Silesian University of Technology
Faculty of Automatic Control,
Electronics and Computer Science

Doctoral Dissertation

Solutions for selected problems of demand forecasting based on machine learning methods and domain knowledge

Joanna Badura

Supervisor : dr hab. Marek Sikora, Associate Professor

Gliwice 2023

Contents

1 Introduction **3**

1.1 Objectives of the work . . . 6

1.2 Structure of the thesis . . . 6

2 Demand forecasting – approaches and challenges **8**

2.1 Problems observed in demand forecasting . . . 8

2.1.1 Tasks connected with short-, medium-, long-term demand forecasting . . . 9

2.1.2 Short-term forecasting . . . 13

2.1.3 Medium-term forecasting . . . 16

2.1.4 Long-term forecasting . . . 19

2.2 Historical data and time series – definitions . . . 21

2.3 Analytical methods used in demand forecasting . . . 25

3 Promotions **32**

3.1 Problem statement . . . 35

3.2 Datasets . . . 37

3.3 Modelling effectiveness of promotions . . . 40

3.4 Practical applications of predictive models . . . 50

3.4.1 Price sensitivity simulator . . . 50

3.4.2 Promotion recommendation system . . . 59

3.4.3 Extended promotion recommendation system . . . 66

3.5 Recommendations using survival action rules . . . 74

3.5.1 Theory for Survival Action Rules . . . 74

3.5.2 Induction of survival action rules . . . 78

3.5.3 Survival action rules for promotion task . . . 82

3.6 Conclusions . . . 91

4 Top-down forecasting **94**
4.1 Problem statement . . . 95
4.1.1 Datasets . . . 95
4.2 Proposed approaches . . . 97
4.2.1 Naive approach (data cube approach) . . . 97
4.2.2 Nearest neighbour (KNN) . . . 99
4.2.3 Linear mixed model (LMM) . . . 103
4.2.4 Ensemble approach . . . 105
4.3 Comparing the proposed approaches . . . 105

5 Incorporating additional time series related to demand forecasting **109**
5.1 Incorporating data from similar locations . . . 110
5.1.1 Datasets used in experiments . . . 117
5.1.2 Results . . . 126
5.2 Using data from the same location . . . 132
5.2.1 Used datasets . . . 138
5.2.2 Experiments . . . 143
5.2.3 Results . . . 147
5.2.4 Analyse of made mistakes . . . 150
5.2.5 Discussion . . . 158
5.3 Conclusions . . . 159

6 Returns **161**
6.1 Problem statement . . . 163
6.2 Linking purchase transaction with return transaction . . . 164
6.3 Creating forecasting model for returns . . . 167
6.3.1 Approach number 1 – using only original attributes . . . 169
6.3.2 Approach number 2 – feature engineering . . . 169
6.3.3 Approach number 3 – parameter tuning . . . 170
6.3.4 Results . . . 172
6.4 Discussion . . . 175
6.5 Conclusions . . . 176

7 Summary **178**
7.1 Future directions . . . 180

1 Introduction

Demand forecasting is an issue that is important in almost every industry that offers products or services to customers. It will apply to commercial vendors, but also to suppliers of basic utilities like water, energy and gas. Demand forecasting will be an attempt to predict the behavior and needs of current or potential customers. On a small scale, such forecasting will be done only on the basis of domain knowledge and one's own predictions. For example, a local beauty salon is able to determine what services it offers, and therefore what equipment or products it will need to provide them. The person in charge of procurement is also able to predict which products may run out, and order them from the manufacturer on an ongoing basis. Such forecasting will be done on the basis of the responsible person's domain knowledge and jugmental forecasting. It can also make use of knowledge of historical transactions. Such a person will, in some way, use historical knowledge for demand predictions – whether in the form of their own memory of past needs or perhaps using transaction history stored in service premises systems.

The challenge comes when demand forecasting is no longer so simple for one person to perform. Such a situation may occur when the scale of the enterprise increases, when many products/services areoffered, or when forecasts must be mage for many locations (e.g. for a chain of stores). Another difficulty may be the horizon of the forecast. There might be the need to forecast demand well in advance in order to order the necessary products, or the other way around – when it is necessary to forecast almost for the nextmoment, where we have a short time to analyse the needs and make the decision. Another factor where forecasting demand may not be a trivial task will be a situation where many internal and external factors affect demand. For example, having one location in the area that offers a particular service and knowing the number of our customers who, on average, come in every month for a particular service, we are able to roughly predict sales in the following month. However, it may be different if there are many similar establishments offering the same service or product, and customers are willing to change vendors if they offer better terms. In such a case, the number of sales can change significantly due to the price offered to us and the competition, advertising or promotions offered. Sales can also be affected by factors beyond our control, such as the time of year, day of the week, holidays, outside temperature or weather. It is worth noting that some of these factors are known in the future (e.g., holidays or our proposed promotions), but some are impossible or difficult to predict (competitor's offer or weather in the distant horizon).

Incorrect and inaccurate forecasts can have significant consequences for a company. First, the entire supply process can be negatively affected, resulting in shortages or delivery delays. Overestimating demand can result in ordering or producing too many products. Underestimating can be even more problematic because it will result in supply shortages, which can result in dissatisfied customers, who will form a bad opinion of our company. Understanding and accurately forecasting product demand is crucial for businesses, as it impacts inventory management, supply chain efficiency, and customer satisfaction.

As it can be seen, demand forecasting in many cases is a complicated issue. Methods based on jugmental forecasting and expert decisions are still being used [104], but more and more, especially in large-scale cases, artificial intelligence methods are being used for the task. Machine learning, a subfield of artificial intelligence, offers many methods that can be used for demand forecasting. Such methods are based on data and can draw patterns from it. For this reason, they are useful in cases where the forecasted value is affected by many factors and the patterns influencing demand are very complex and difficult to discover manually.

When thinking about demand forecasting, the first thing that may come to mind is the problem of forecasting sale a product that has already been on sale, that is, for which we have historical data. This is probably the most typical, and thus repeatedly described in the literature, demand forecasting issue. For example, if a grocery store has apples on sale on a regular basis, it is obvious that the shop will generate on an ongoing basis demand predictions for the apples. The issue of sales forecasting for known products will be applicable to almost any company that offers something. Sales of a given product can be presented in the form of a time series. This means that for specific points in time we are able to assign corresponding sales. In this case, we can use forecasting methods typical for time series but we are not limited only to them. Machine learning offers method, each with its strengths and drawbacks. For demand forecasting we can use simple method like forecasting based on last known value, moving average or linear regression. Their advantage is certainly the simplicity of implementation and understanding. We could also use more statistical methods like Autoregressive Integrated Moving Average (ARIMA) [29], Seasonal ARIMA (SARIMA) [29], hybrid versions of ARIMA [179, 79, 163] or exponential smoothing [63]. Another group of machine learning methods are tree-based methods like Decision Trees [32], Random Forest [31] or Gradient Boosted Trees (GBT) [37]. The advantage of Deep learning models [103, 133, 181] its their ability for capturing long-term dependencies in time series data. Machine learning methods that can be used for time series forecasting are certainly not limited to the methods mentioned above.

However, it can be noted that in the context of demand forecasting one can come across other issues that are not typically related to time series forecasting. For example, one may wonder how to forecast demand for a new product that has no historical sales data. On the other hand, we may have a lot of historical data that comes from one or more locations, and we may want to use it to improve the accuracy of forecasts. Often, product sales are affected by factors that can be deliberately introduced and controlled by companies, such as price changes, promotions and additional offers. Their impact is very important on the final sales of the product under study and co-sold products. In addition to typical sales, we also have other product-related events, such as complaints and returns, which can also be forecast. Another aspect is also how to influence sales in such a way as to accelerate or delay the sell-out of a given product.

As we can see, the demand forecasting issues are not limited only to forecasting demand for products based on existing historical sales data for a specific product. The topic of demand forecasting is much broader and a given dissertation will devote attention to different aspects connected with this topic. Some of the aspects addressed were identified as a research gap in context of machine learning methods usage.

The application of machine learning in demand forecasting goes far beyond just choosing the right algorithm. The first important aspect is available data and its quality. To do this, it is necessary to prepare the data properly. It is important to create appropriate conditional attributes to describe relevant phenomena. Some of this data can be added using external databases (e.g., information about the weather or vacations and national holidays), but sometimes there is information that can only come from a particular enterprise. In the case of such gaps, it is important to artificially model such information. Another aspect that is important to consider is domain knowledge. Certain behaviors and phenomena can only be explained on the basis of specialized knowledge. It can also be important for the proper formulation of the target variable and specifying the performance metric. Important is also to create models that can be understood by the user, if required. More and more solutions are being created for explainable artificial intelligence (XAI), which can be used to interpret model decisions or describing in which situations predictive model makes mistakes. There are many such aspects to take into account and they may vary from issue to issue.

The dissertation presents the author's breakdown of demand forecasting issues. Subsequently, selected issues are discussed in more detail. The research used real data from real companies belonging to different industries. Often these are collections covering sales data from several years and for multiple products. Due to their sensitive nature, they are proprietary.

The author of the dissertation is a holder of European Unionscholarship through the European Social Fund, grant InterPOWER (POWR.03.05.00-00-Z305). The work concerned falls under Artificial Intelligence and Data Processing Priority Research Area of Silesian University of Technology.

1.1 Objectives of the work

The first aim of the thesis is to propose a breakdown of demand forecasting issues that can be solved by machine learning methods. The next objective is to propose the use of machine learning methods and domain knowledge for selected issues. The selection of these issues was determined by carried out research projects.

1.2 Structure of the thesis

This thesis consists of 7 chapters. Chapters 1 and 7 are respectively the introduction and summary of the thesis.

Chapter 2 is an introduction to the topic of demand forecasting using machine learning and domain knowledge. Section 2.3 presents the author's breakdown of demand forecasting issues. The issues are subdivided according to the forecast horizon and the time factor. Each issue is described in a separate section and possible problems are identified for each. Section 2.2 presents the theoretical issues involved in forecasting, i.e. historical data and time series. Section 2.3 provides a brief overview of machine learning methods that can be used in the field. This subsection is certainly not exhaustive, but neither is the purpose of this work.

Chapter 3 presents the first selected issue related to demand forecasting, namely focusing on the topic of forecasting promotional effectiveness. A literature review for the topic is presented. In Section 3.1, the problem statement is presented and 6 promotion effectiveness indicators are proposed, which were used to forecast the effectiveness of promotion. Section 3.2 presents the data used for the problem statement and Section 3.3 shows how promotion effectiveness is modelled. Section 3.4 presents practical applications of the forecasting models used and presents a price sensitivity simulator and a system for recommending promotions. T he last section of this chapter – 3.5 – presents an innovative method for the induction of Survival Action Rules and shows how these rules can be used to find changes to improve the effect of promotions.

Chapter 4 focuses on the issue of demand forecasting and, in particular, on the problem of top-down forecasting, i.e. the breakdown of a higher-level forecast (e.g. for a group of products) into a lower-level forecast (e.g. a forecast for a specific product). The chapter outlines the statement problem, the datasets used, presents the methods investigated and compares the approaches investigated for breaking down higher-lvel forecasts into individual product forecasts.

Chapter 5 presents the next issue highlighted in the breakdown, namely the manner and effectiveness of using additional time series to improve the forecasting of the phenomenon under investigation. Subsection 5.1 focuses on the problem of adding data from other locations describing a similar issue, and subsection 5.2 focuses on adding data from the same location using data from the digital-twin model of a building as an example.

Chapter 6 focuses on the issue of forecasting the probability of return of a specific product. Section 6.1 presents the problem statement. Section 6.2 presents a proposed way of combining a purchase transaction with a corresponding return transaction when there is no such data in the database. The given approach allows the creation of forecasting models that predict the return probability based on information about the purchase and the circumstances of the purchase. This is outlined in subsection 6.3. Subsections 6.4 and 6.5 provide a discussion and a sub-structure of the chapter in question.

2 Demand forecasting – approaches and challenges

Forecasting is a task of predicting unknown values based on available knowledge. In many cases, predictions are made for values in the future. The knowledge used in forecasting could be historical values of predicted phenomena gathered in spreadsheets or databases, but it could also be the experts' knowledge that had never been written down in any way. It is their experience that can give them the ability to make good predictions.

Demand forecasting is no exception in this case: to make good forecasts, the prediction system or expert (retailer) needs to base on knowledge gathered from the past. In the simplest case, they would make predictions of, e.g. demand for a product in a store based on historical sale of this specific product in the store. This would be the task of time series forecasting. However, demand forecasting in many cases is a more complex task than this.

In this work, by demand forecasting we mean forecasting the future values based on past sale data but also predicting consumer demand for products in the future. In the literature we can find two terms for these forecasting: the first is called *sales forecasting* and the latter – *demand forecasting*. In this work, *sale* and *demand* terms will be used interchangeably and *demand forecasting* and *sale forecasting* will be mean the same concept.

In the first part of this chapter, some challenges that can occur when working on demand forecasting will be addressed.

2.1 Problems observed in demand forecasting

Sale forecasting, based on historical data, is only one of the issues related to demand forecasting. I would hazard a statement that this is the task that is the best known and developed from all the tasks encountered in demand forecasting. However, there are more challenges that are equally important and there is a need to address them. They depend on the level of company development, sale strategy, types, and diversity of sold products. In this chapter, I want to outline the range of topics related to demand forecasting.

The figure 2.1 presents a proposed in this work novel breakdown of challenges and tasks connected with demand forecasting. They will be addressed in details in the followed subsections.

I proposed to divide the cases in demand forecasting into three categories: cases connected with short-term forecasting, medium-term forecasting and long-term forecasting. This breakdown is dictated by various business cases that allow to introduce such a division. The time ranges of the horizons in question can vary from one company to another. For the purpose of this paper, I have marked a short-term as forecasting for one week ahead, medium-term as 2 weeks to 3 months ahead and long-term as forecasting for more than 3 months ahead.

2.1.1 Tasks connected with short-, medium-, long-term demand forecasting

Some tasks are the same for short-, medium- and long-term forecasting. Also, no significantly different problems are encountered for the task. Due to this fact, the followed tasks will be presented jointly for all forecasting horizons.

Forecasting demand when starting new business

Forecasting the sale for all products when launching new business is a very specific problem of forecasting. In this situation, probably only market surveys can help to forecast the probable sale of products. The retailer can also base on data from different, similar brands, however the access to this data in most cases would not be possible. In this case, judgmental forecasting can be used. Problems connected with this case:

- Lack of any historical data.
- Lack of access to data from similar industry/business.

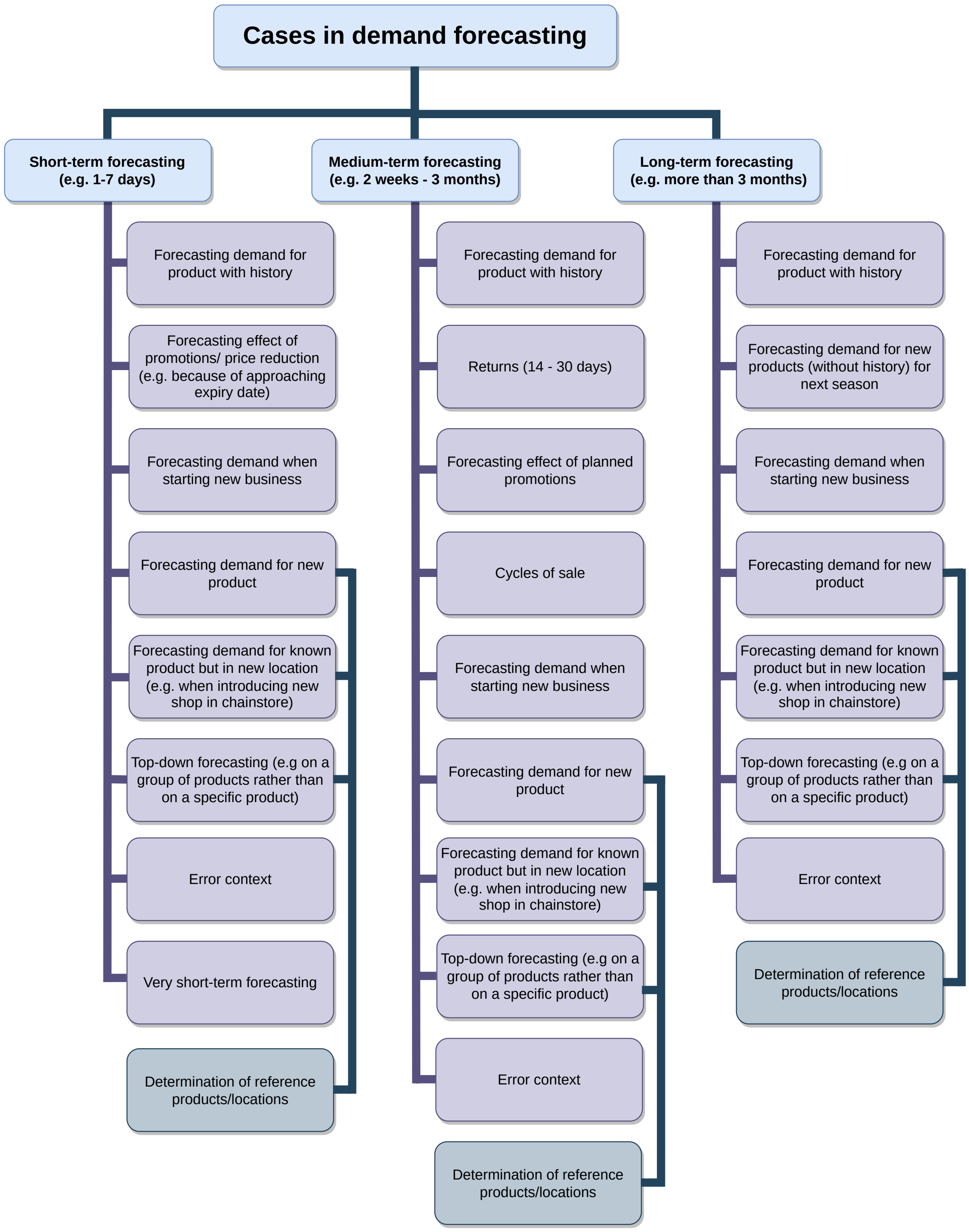

Cases in demand forecasting
Short-term forecasting (e.g. 1-7 days)
Forecasting demand for product with history
Forecasting effect of promotions/ price reduction (e.g. because of approaching expiry date)
Forecasting demand when starting new business
Forecasting demand for new product
Forecasting demand for known product but in new location (e.g. when introducing new shop in chainstore)
Top-down forecasting (e.g on a group of products rather than on a specific product)
Error context
Very short-term forecasting
Determination of reference products/locations
Medium-term forecasting (e.g. 2 weeks - 3 months)
Forecasting demand for product with history
Returns (14 - 30 days)
Forecasting effect of planned promotions
Cycles of sale
Forecasting demand when starting new business
Forecasting demand for new product
Forecasting demand for known product but in new location (e.g. when introducing new shop in chainstore)
Top-down forecasting (e.g on a group of products rather than on a specific product)
Error context
Determination of reference products/locations
Long-term forecasting (e.g. more than 3 months)
Forecasting demand for product with history
Forecasting demand for new products (without history) for next season
Forecasting demand when starting new business
Forecasting demand for new product
Forecasting demand for known product but in new location (e.g. when introducing new shop in chainstore)
Top-down forecasting (e.g on a group of products rather than on a specific product)
Error context
Determination of reference products/locations

Figure 2.1: Breakdown of demand forecasting issues.

Forecasting demand for new product

One of the biggest challenges is the issue of forecasting the demand for new product. It can be a completely new product or product similar to other so far sold in the shop. Having no data (no historical time series), we need to find a way how to forecast the demand, because it is a crucial factor when ordering products from suppliers.

When a completely new product is launched, we need to rely on experts assumptions for the future. However, in most cases, the retailer already has more or less similar products in the product range.

The first challenge is how to determine similar products. In some cases it is simple – e.g. we already sale different brands of butters, so sale of a new butter will probably behave similar as the others. More challenging may be finding similar products, e.g. in fashion business. When a new model of trousers is introduced, it can be difficult to decide if the sale will behave similar to all trousers, or maybe only these that are similar in shape or colour. The problem of grouping products is introduced.

Another difficulty when introducing new product is the aspect of sale cannibalism. Continuing example of selling butter – the demand for butter among clients is rather fixed. Introducing a new butter brand will affect the decline in sale of other brands.

Problems connected with this case of demand forecasting:

- No demand history of the product.
- Lack of similar products or inability to identify a similar product.
- Sale cannibalism of products sold so far by a new product.

Forecasting demand for known product but in new location

When we forecast the demand for chain stores, it can happen that a new shop is launched. The shop may sell the same products as in other shops, but the sale won't be the same as in all shops. Because of the localization, the sale of different products can be similar among one shops and very different from others. The differences may come e.g. from different climate, wealth of people that live in a surrounding, the approximation of shops from other brands or the fact whether the shop is located in a tourist destination. The knowledge about a sale from different stores is crucial when launching a new store, but it is important to understand which shops are similar and which products will sell specifically only in some groups of stores. The example can be a sale of bread and a sale of caviar – sale of bread will probably be similar in all locations in one country, when caviar, as a luxury good, may be sale better in more prosperous regions.

Similar problem is observed not only for new shops in a chain store. It always occurs when new localization is introduced. For example, when forecasting energy demand in many households and new building has been connected to the network. In this situation, we already have historical data from different locations, and we want to use it in order to predict energy demand for a new household.

Problems connected with forecasting for new location are as follows:

- Inability to identify a similar location.
- Need to identify the seasonality of the location (e.g. stores at the seaside or in the mountains may require unique treatment due to their seasonality).

Top-down forecasting

Top-down forecasting can occur in any industry in which products can be grouped at different levels of granularity. This kind of forecasting may be used when demand for a product is very low and rare. In this case it is difficult to properly, on an acceptable level, forecast the demand for this product. Because of this, it can be helpful to forecast the demand for a group of products and then the grouped forecast split into forecasts for specific products. The problem of such forecasting is losing information about unique characteristic of products and their prices because they are not included in forecasting model. However, the advantage is a possibility to improve model quality as a result of a higher level of aggregation.

Problems observed in top-down forecasting:

- Development of a way to break down the forecast of a higher level to a lower level.
- Finding impact of products different characteristic interactions on the probability of demand. For example, how the combination of features of a product in a given colour, size, and price will affect the percentage of total sales that the product accounts for.

Determination of reference products and locations

When forecasting demand for a new location, a new product or a product that is rarely in sale, it can be desired to determine the reference product/location for this one. In this case, the demand predicted for a referenced product/location can be used as a prediction for the concerned one.

The biggest problem in this case is lack of data with very specific characteristics. Because of insufficient amount of data describing products, some of them may wrongly seem alike. For example, when forecasting the fuel sales at gas stations, we may indicate that two gas stations in the same area should have similar sale. However, lacking the information that next to one of them there is highway, can give wrong conclusion about reference location. Another example, this time for fashion sector: two blouses in this same colour, size, and model should have similar characteristic of demand. However, without information about overprinted graphic or quality of workmanship, the referenced products may be incorrectly indicated.

Error context

Specific business conditions, and thus the importance of errors, can be difficult to capture by standard error measures used in forecasting models. This same value of a forecasting error may have a different meaning in different situations. For example, for a specific combination of characteristics of sale period and for a specific combination of characteristic of a product, even a small error may have big meaning when in different situation it would be negligible and absolutely acceptable. The case is how to include this knowledge in the process of forecasting model preparing. Problems connected with this case of demand forecasting:

- The domain expert is unable to indicate logical conditions when error changes its default meaning.
- Time-consuming process of reviewing and labelling each observation in historical data.

2.1.2 Short-term forecasting

Demand forecasting for a product with a history

Forecasting demand for products with history is the most common cases in demand forecasting, and the one that probably first comes to mind when thinking about this domain. It means that the product or medium was already in sale in the specific location, and we have the history of the demand. This forecasting task has several problems that can be encountered:

- Short history of demand. It means that for the product we do not have complete information about how the product behave, e.g. throughout the whole year or during special holiday periods.
- Predicting demand for numerous products at a same time.
- Forecasting demand for the same product but in many locations.
- Time series classification of the response variable (smooth, erratic, etc.) – this may lead to a custom-made model deployment per recognized class, such as:
  - High-volume with rare sales
  - Low-volume with rare sales
  - High-volume with frequent sales
  - Low-volume with frequent sales
- Trend recognition on the response variable.
- Forecast extrapolation outside the range of historical sales recorded on the training data.
- Forecast and model assessment — the properly selected metrics following the customer's needs. For some, product sales distribution is a priority, not the forecast quality.
- Price sensitivity -– when experiencing high inflation levels, the selling prices should be adjusted to the CPI (Consumer Price Index); in other words, the model needs to follow the *real*, not the *nominal* prices.
- Seasonal character of the sales points (stores at the seaside or in the mountains may require unique treatment, it is due to their seasonality).
- Product cannibalization (an increase in demand for one product reduces demand for another).

Forecasting effect of promotions and price reduction

This task may be connected with forecasting the demand when the price will be reduced due to the fact that the product has approaching expiry date. Issues connected with this task are as follows:

- Product was never in promotion or with price reduction.
- Low price variation within one product in historical data.
- Proper recognition of the promotional sales.
- Length of the marketing actions – overextended promotions are ineffective and should not be treated as promotions in modelling.
- Multipacks – for example: buy three, one is free.
- Forecast and model assessment – the properly selected metrics following the customer's needs. For some, product sales distribution is a priority, not the forecast quality.
- Price sensitivity – when experiencing high inflation levels, the selling prices should be adjusted to the CPI (Consumer Price Index); in other words, the model needs to follow the *real*, not the *nominal* prices.
- Product cannibalization (an increase in demand for one product caused by promotion reduces demand for another product).

Very short-term forecasting

It is a special kind of short-term forecasting. In this case we want to make predictions "for the next moment", e.g. for the next 5 minutes, next hour or next 24 hours. The challenges of this case of short-term forecasting are:

- Forecast calculations must happen relatively quickly.
- When making a forecast for the next "moment", we may not have information about the actual consumption from the moment before the given forecast period. For example, if we are doing a forecast for the next day and for business reasons we need to have a given forecast at 15:00, we only have information from today's actual demand up to 15:00 and not the actual demand from the whole day. For this reason, our forecast is burdened by the lack of information on the actual total consumption from the previous period.

2.1.3 Medium-term forecasting

Demand forecasting for a product with a history

This task of demand medium-term forecasting is analogous to the same case described in short-term forecasting subsection. The only difference is the time in the future for which we want to predict demand. The problems connected with this task are:

- Short history of demand. It means that for the product we do not have complete information about how the product behave, e.g. throughout the whole year or during special holiday periods.
- Predicting demand for numerous products at a same time.
- Forecasting demand for the same product but in many locations.
- Time series classification of the response variable (smooth, erratic, etc.) – this may lead to a custom-made model deployment per recognized class, such as:
  - High-volume with rare sales
  - Low-volume with rare sales
  - High-volume with frequent sales
  - Low-volume with frequent sales
- Trend recognition on the response variable.
- Forecast extrapolation outside the range of historical sales recorded on the training data.
- Forecast and model assessment – the properly selected metrics following the customer's needs. For some, product sales distribution is a priority, not the forecast quality.
- Price sensitivity – when experiencing high inflation levels, the selling prices should be adjusted to the CPI (Consumer Price Index); in other words, the model needs to follow the *real*, not the *nominal* prices.
- Seasonal character of the sales points (stores at the seaside or in the mountains may require unique treatment, it is due to their seasonality).
- For medium-term forecasting, a long to a very long forecast horizon may occur.

- Product cannibalization (an increase in demand for one product reduces demand for another product).

Returns

Returns of products is a phenomenon that occurs simultaneously with the sale of products. Their effect is especially important in e-commerce sector. Returns have an impact on a demand, and they reduce the effect of sale. On the other hand, companies, due to competition, are increasingly extending the return period, which makes it even more difficult to predict the final demand for products. Because of this, the forecasting not only sale, but also the returns of products, is important to properly stock up products. It is critical to predict, which products or orders are more likely returned. This is a relatively new challenge, as the growth of e-commerce sales was especially noticeable during the pandemic period in 2020. The standard period allowing for the return of products is 14 to 30 days, which is why the problem has been classified as a medium-term forecasting task.

Problems observed when dealing with task of forecasting returns:

- Inability to unambiguously link a return transaction with a purchase transaction.
- Probability calculation based on the reference product may be misleading.

Forecasting effect of planned promotions

Task of forecasting effect of planned promotions is similar to task of forecasting effect of price reduction in case of e.g. approaching expiry date. However, few differences can be pointed out. In case of price reduction in short-term horizon, the aim is to sale all products because otherwise it will go to waste. In case of planned promotions, the price reduction may have multiple reasons: good promotion can increase sales of promoted product, but it can be also introduced because it can attract more clients to the shop or the promotion can encourage to sale multiple items together with the promoted product. Planning and forecasting effect of promotions are more business cases. Problems of forecasting the effect of planned promotions are as follows:

- Product was never in promotion or with price reduction.
- Low price variation within one product in historical data.

- Proper recognition of the promotional sales.
- Length of the marketing actions – overextended promotions are ineffective and should not be treated as promotions in modelling.
- Multipacks – for example: buy three, one is free.
- Forecast and model – the properly selected metric following the customer's needs. For some, product sales distribution is a priority, not the forecast quality.
- Price sensitivity – when experiencing high inflation levels, the selling prices should be adjusted to the CPI (Consumer Price Index); in other words, the model needs to follow the real, not the nominal prices.
- Product cannibalization (an increase in demand for one product caused by promotion reduces demand for another product).

Product life cycle

For many products we can distinguish 4 stages of demand: introducing to the market, sales growth, saturation (maturity) and decline. These four stages create a cycle of sale. When sealing product, we want to recognize each stage well in order to properly manage and maximize profit with desired sales. After recognition, we also want to manipulate the change of a cycle when some undesirable deviation from the assumed cycle is happening. The action may be crucial to carry out when the cycle is predicted to happen too fast (the product will be sold out before the end of a season so it won't be available to the clients) or the sale cycle is changing too slow (at the end of a season not all units of product will be sold). We want to be able to manipulate the cycle in order to go out at the zero level in the warehouse.

Problems that may be encountered when working on this task:

- The occurrence of different product life cycle patterns for various products or product categories.
- Inability to determine the ideal pattern of life cycle for specific category of products to which all product from the category will be compared to.
- The problem of how to manipulate product sale in order to match the ideal life cycle pattern.

2.1.4 Long-term forecasting

Demand forecasting for a product with a history

This task is analogous to the same case described in short-term forecasting and medium-term forecasting subsection. Some differences can be observed when describing problems connected with this task:

- No sales history for the whole year.
- Predicting demand for numerous products at a same time.
- Forecasting demand for the same product but in many locations.
- Time series classification of the response variable (smooth, erratic, etc.) – this may lead to a custom-made model deployment per recognized class, such as:
  - High-volume with rare sales
  - Low-volume with rare sales
  - High-volume with frequent sales
  - Low-volume with frequent sales
- Trend recognition on the response variable.
- Forecast extrapolation outside the range of historical sales recorded on the training data.
- Forecast and model assessment – the properly selected metrics following the customer's needs. For some, product sales distribution is a priority, not the forecast quality.
- A limited number of features based on which the model deploys.
- A high level of data aggregation automatically smooths time series.
- A very long forecast horizon (up to one year).
- The unpredictable external environment being an outsider to the modelling.
- Increased risk of forecast failure (the further it goes into the future, the fewer chances of success).
- Product cannibalization (an increase in demand for one product reduces demand for another product).

Demand forecasting for a new product with a seasonal sale

A given problem is observed if a group of products is not sold continuously, but only within a strict time frame. The products are in sale only in some specific period of time and they are never coming back. In their place, new products are introduced. They may be similar to the just removed ones or completely different in style or features. As a result, we operate using many short time-series. This problem is most common in fashion industry where products are sold only in one fashion season. Additionally, in this sector another challenge is that products are sold in fashion collections. It means that many new products are introduced at the same time, and they replace most of existing products. Very often new products' collection is based on the long-term vision of the creator and this is very difficult to capture in the data.

- Almost all products are new (e.g. in fashion industry).
- Changing customer preferences, changing trends.
- Products described by a few features – difficulty to find a similar product in history.
- Demand forecasting for this task is primarily concerned with forecasting production, not with stocking shops with products already produced (e.g. most production in the fashion industry is done in China a year in advance, and it is difficult to "predict" what the fashion trends will be in a year's time).
- A limited number of features the model deploys.
- Subjective or an expert-based similar product selection as a base for further research in modelling.
- A high level of data aggregation automatically smooths time series and shortens the scalar of data.
- A very long forecast horizon (up to one year).
- Increased risk of forecast failure (the further it goes into the future, the fewer chances of success).
- A hard-to-capture vision for the entire clothing/product collection, which may be completely different from the previous ones.

Several of the issues highlighted were selected for further research. The following chapters of the work focus on the topics of:

- forecasting effect of planned promotions,
- top-down forecasting,
- incorporating additional time series related to demand forecasting (sub-topic of demand forecasting for a product with history),
- returns.

The topics were selected because of the R&D projects underway.

2.2 Historical data and time series – definitions

So far, the term of historical data was used. It means that we have data describing the past demand and it can be stored in different forms – in a structured manner (e.g. historical values of sales stored in spreadsheets), in many documents and systems or it can be a knowledge of an expert gained from past experience.

When thinking about historical sales data, a natural form of presentation for such data is a time series. In most cases, if a company records its sales data or consumption over time, this is the form of data we will get.

A time series is a sequence of data points (observations) taken sequentially in time [28]. Each instance of a data set represents a different time step, and each attribute contains a value connected with that time. Depending on the dataset, observations can represent regular samples. However, if it is not the case, the attribute *timestamp* must be included in the data[168]. In most cases it will indicate the time when the observation was obtained, but it can also describe the time span when some phenomenon occurred (e.g. observation marked with date '1995-09-30' can describe the total sale that was obtained during whole September 1995).

In the work, I will represent each observation of time series as

$$\{X_t, t \in T\} \tag{2.1}$$

where t is a timestamp of the observation that belongs to the set of times T.

The simplest graphical representation of a time series is a plot, where on axis X is time and axis Y presents the values of our interest corresponding to the timestamps

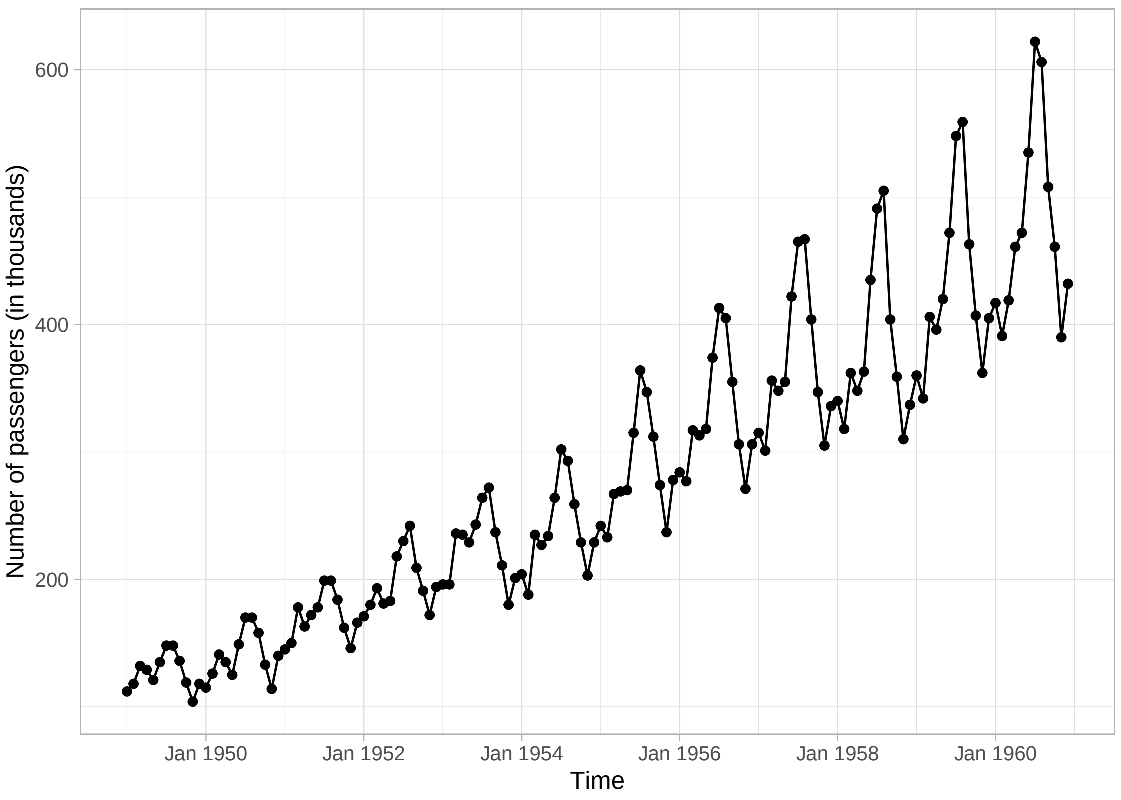


Figure 2.2: Exemplary plot for time series representing monthly airline passenger numbers, 1949 to 1960.

[81]. The example of this plot is visible on Figure 2.2, which represents the time series dataset describing monthly totals of international airline passengers, 1949 to 1960. The dataset is publicly available in R `datasets` package and was described in [29].

Time series can be of different types, what can have an impact on further analysis and forecasting model used for predicting future values of time series. I will mention some of possible time series classifications proposed in [33]:

- We can classify time series by the time between observations. If the observations are made in equal intervals of time, than this time series is contiguous. If the observations are not uniform over time than it is a discontiguous time series. It can happened due to corrupted data (lack of data) or by a specific domain of the time series (e.g. observations are measured by human in some specific situations).

- Second classifications is based on number of measured variables that are the part of time series. If only one variable is measured over time (e.g. temperature in the room), then it is an univariate time series. Many variables measured over time create multivariate time series.
- We can also divide time series to unstructured and structured ones. Structured time series would be a time series where trend and seasonal cycles are visible. Unstructured time series would be the one where there is no obvious patterns in the data.

When describing classification of time series, I mentioned the patterns that can be visible in time series. Based on [81] we can distinguish:

- *Trend* – it is a tendency of increase or decrease in the data. It can change over time, e.g. the sale of the product can be increasing over time and after some time the sale can start to decline. The timestamp where the trend changed can be also named as *changepoint*. The changepoint detection is another problem in data science.
- *Seasonal pattern* - is a pattern connected to time with a fixed and known period, e.g. data can be affected by the month or weekday of the observation. For example we could observe seasonal pattern in the sale of ice-cream because sale will be higher than the average sales always during the summer months.
- *Cycle* - these are the fluctuations visible in the data that are not of fixed frequency (unlike seasonal patterns). Cycles are mostly connected to some business or economic conditions.

In the time series we can also observe a random fluctuations, that can be also named as *reminder of the time series*. It can happened that time series has no trend, cycle or seasonal pattern but only this random fluctuations.

It is common to decompose time series into several components described above. It can help to understand the patterns in time series. In the decomposition process, it is common to split time series to three components: *trend-cycle* component (often called *trend* component), *seasonal* component and a *reminder* component [81]. Depending on the period of observations, several seasonal components can be distinguished, corresponding to different seasonal periods. For example, sale of ice creams can be

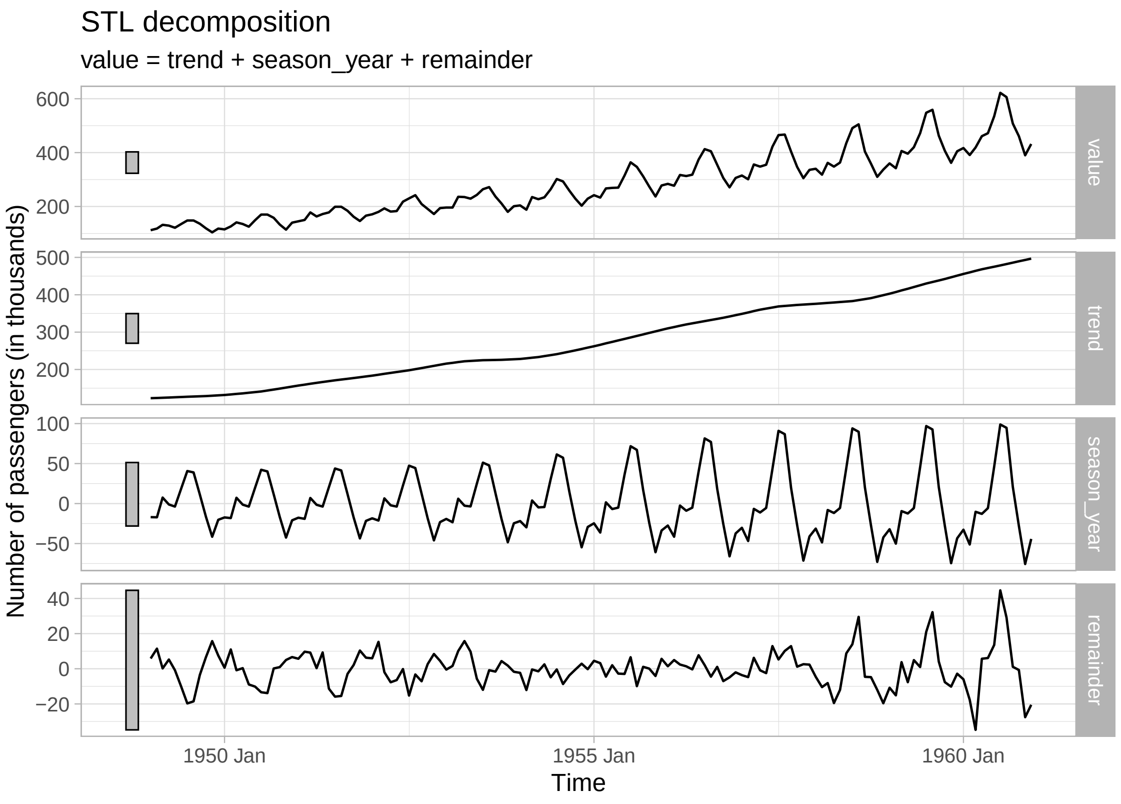


Figure 2.3: Decomposition of time series representing monthly airline passenger numbers, 1949 to 1960.

changing based on weekday but also during whole year the sale is different in different months and this pattern is repeating each year.

The decomposition of time series describing monthly airline passenger numbers is presented in Figure 2.3. The decomposition was perfomed using STL method ("Seasonal and Trend decomposition using Loess") described in [42].

However, as I mentioned before, demand forecasting is not only forecasting based on historical sale data an its trends. It means that demand forecasting problems can not be all solved using only methods typical for time series forecasting. In the next section I will describe some groups of methods that can be used in demand forecasting tasks.

2.3 Analytical methods used in demand forecasting

When describing types of forecasting methods, we can classify them based on different factors. In [93] authors have proposed the division for weather forecasting systems but I think proposed classification to a large extent is also appropriate to demand forecasting. Division of forecasting methods can be made based on number of analysed variables and in this case we can distinguish univariate and multivariate forecasting methods [53, 138]. Next, we can classify forecasting methods based on duration of the forecasting. In section I have presented different cases of demand forecasting with this division – for short-term, medium-term and long-term forecasting. Another division can be made based on number of time steps that need to be predicted. We can also distinguish different methodologies that can be used in order to perform demand forecasting.

The first approach is *judgmental forecasting* [66, 104]. It can be performed by experts with no additional help of decision support system or prediction model. Sometimes simple baseline statistical forecast with a judgmental adjustment [56, 135] is used. It means that the planning process is often done manually. However, studies have shown that using only these kinds of forecasting methods may bring bias [112, 15]. A better idea may be to use more advanced methods that rely mostly on knowledge that comes from historic data stored in more structured way. For this, statistical and artificial intelligence methods can be used. It is not possible to present all available methods that could be used for demand forecasting task. There are several review works that focus on this topic [83, 20, 137, 156, 5]. However, I will focus on the most important ones or the methods mentioned in further parts of this doctoral dissertation.

When talking about time series forecasting, there is two methods that are probably the most known and are the most established ones – it is Autoregressive Integrated Moving Average (ARIMA) [29] and exponential smoothing [63]. Both of them, we would classify as a statistical approaches for the task of time series analysis. ARIMA model focus more on describing the autocorrelations in the data, while exponential smoothing aim to describe trend and seasonality in the data [81]. ARIMA is a combination of differencing with autoregression and moving average model. The variation of this model is SARIMA - seasonal ARIMA model [29].

Another method that we would classify as statistical method is Prophet [2]. It can be treated as an extension of ARIMA approach. Prophet is an open-sourced library created by Facebook in 2017. Its goal is to analyse time series data and forecast its future values. The Prophet models take into account trends, seasonality and holidays. Multiple seasonality can be included. The Prophet model can be represented as decomposed time series, and they can be described with the following equation

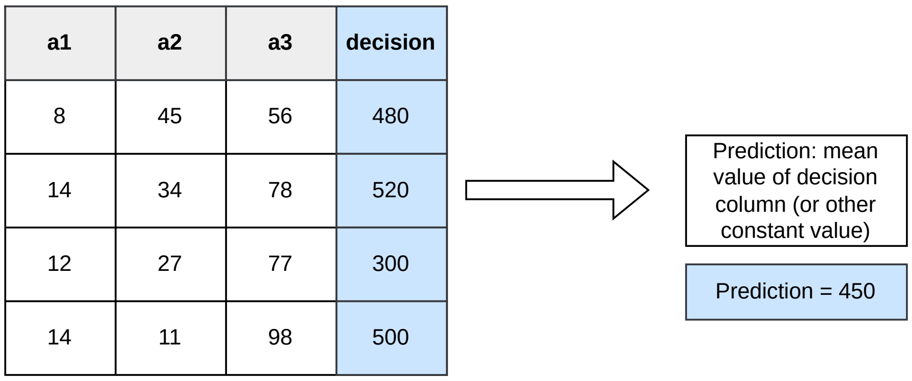
$$y(t) = g(t) + s(t) + h(t) + \epsilon_t \tag{2.2}$$

where $g(t)$ represents trend function, $s(t)$ represents seasonality (daily, weekly, yearly), $h(t)$ represents the holiday effect and ϵ_t is the error.

The important feature is that the library is possible to use without expert knowledge about time series forecasting. The first results can be obtained without setting any parameters. The user can provide only historical data of a time series in an adequate data format, and they will obtain the forecast of future values. Thanks to this feature, it can be very useful for users without data science background. On the other hand, the Prophet library is highly customizable, so the more experienced user can add much additional information, that can be useful for the model e.g. information about special events that can have impact on demand. An in-depth description of the Prophet algorithm can be found in the paper [149]. The authors of the initial Prophet paper have shown that Prophet models can outperform popular automated forecasting techniques like Auto-ARIMA and TBATS.

Another approach to demand forecasting is using methods connected with tabular data. One method that is worth mentioning in this place is the gradient boosting algorithm [58] and one of its implementation – the extreme gradient boosting (XGBoost) algorithm introduced in [37]. Gradient boosting is an ensemble algorithm which create a prediction model in the form of a team of multiple weak learners (mostly decision trees) which when combine produce a strong model. When weak learners are decision trees than the algorithm may be called Gradient Boosting Decision Trees. It is a well known fact that XGBoost, implementation of gradient boosting algorithm, is highly effective for a vast range of classification and regression problems – in medicine [125, 35, 152], in fault detection [154, 177], fraud detection [180, 67], accident detection [127], and many others. The main idea that goes under the XGBoost is to use many weak learners into a single strong learner. In an iterative way, each predictor tries to correct the residual errors of a preceding predictor.

In the default form of the implementation, in the first step of the algorithm all observations are given the same prediction, e.g. a mean value of a decision attribute in train dataset or other constant value. It is shown in Figure 2.4 – for an example



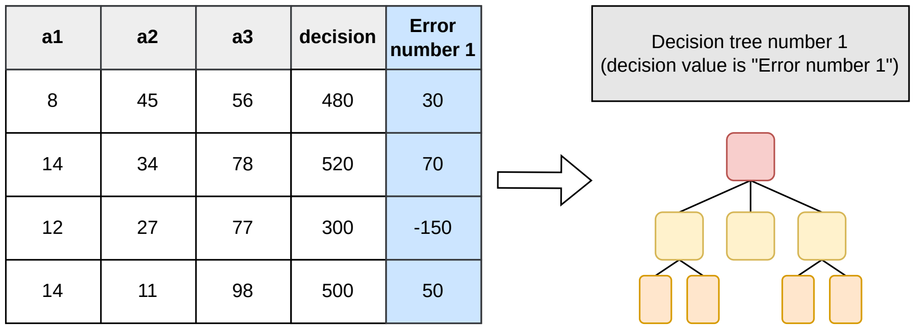
a1	a2	a3	decision
8	45	56	480
14	34	78	520
12	27	77	300
14	11	98	500

Prediction: mean value of decision column (or other constant value)
Prediction = 450

Figure 2.4: Creating first prediction for the dataset in the Gradient Boosting Decision Trees algorithm.

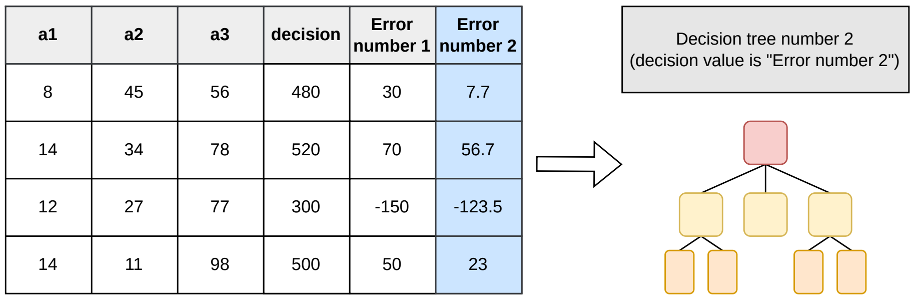
dataset whose conditional attributes are *a1*, *a2* and *a3* and having a decision column with a numerical value, the first prediction is the average of the values in the decision column. The given predictions generate errors (residuals). In the next step, a regression tree is fitted to the data and the learned labels are the errors generated in previous step what is shown in Figure 2.5. Created decision tree is used to obtain new predictions, which mostly are calculated as $F_1 = F_0 + \alpha * F_{T1}$, where F_1 is a forecast for second step of algorithm (after first weak learner), F_0 is an initial forecast (e.g. mean value of a decision column), α is a hyperparameter describing learning rate and F_{T1} is a prediction obtained from the first decision tree. Assuming that learning rate is equal 0.1 and the prediction for the first decision tree for observation number 1 is 223 than the F_1 prediction for observation number one would be calculated as $450 + 0.1 * 223 = 472.3$. The residual for this example would be $480 - 472.3 = 7.7$ and it would be used as a decision value for the second weak learner. Errors obtained from this step are the new fitted labels in the next regression tree. It is presented in Figure 2.6. The process is repeated M times (value M is defined by user) and in the result M weak learners are created. Their combination creates single strong learner – Gradient Boosting Decision Tree model. In the Figure 2.7 is presented the process of obtaining prediction for the new example. The forecast from Gradient Boosting Decision Tree model would be calculated as $F_0 + \alpha * F_{T1} + \alpha * F_{T2} + ... + \alpha * F_{TM}$. Thanks to procedure of creating multiple weak learners, the model can correct each own mistakes.

The problem that can be encountered is the fact that XGBoost is not a typical method used for time series task. In demand forecasting and historical sale data,



a1	a2	a3	decision	Error number 1
8	45	56	480	30
14	34	78	520	70
12	27	77	300	-150
14	11	98	500	50

Figure 2.5: First decision tree (weak learner) in the Gradient Boosting Decision Trees algorithm created based on residuals.



a1	a2	a3	decision	Error number 1	Error number 2
8	45	56	480	30	7.7
14	34	78	520	70	56.7
12	27	77	300	-150	-123.5
14	11	98	500	50	23

Figure 2.6: Second decision tree (weak learner) in the Gradient Boosting Decision Trees algorithm.

the time factor is very crucial because the sale happens in some moment in time. We also want to forecast for the specific time in future. XGBoost do not support this in the objectives of this method. In order to use the XGBoost algorithm, or any other algorithm designed for tabular data, for demand forecasting feature extraction and data preparation is a crucial step. There are special methods that automatically generates features describing time of observation as a set of additional attributes but it can produce many unneeded features that also can be highly non-intrepretable. The better idea may be to create new attributes describing time and relationship between observations close to each other by hand and in thoughtful way.

The last group of algorithms that are highly used in demand forecasting are Artificial Neural Networks algorithms. Deep learning is widely used in time series forecasting problems [21] and there are many architectures that can be used for the task. Probably the simpler neural network used for time series problem can be Multilayer Perceptrons (MLP) [117]. Another architecture worth mentioning is Convolutional Neural Networks (CNN) [107]. This neural network is mostly known

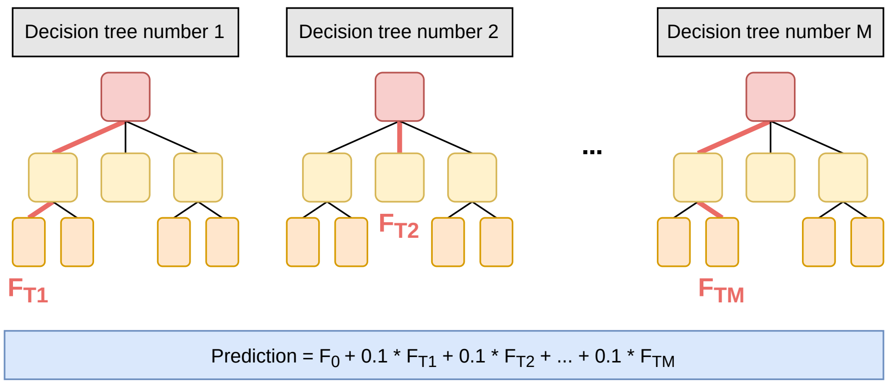


Figure 2.7: Calculating prediction for new example using in the Gradient Boosting Decision Trees model.

for handling image data but it can be also used in time series problem because CNNs has an ability to learn and extract new features from raw input data. A sequence of data (time series) can be treated as one-dimensional image and it can be processed by CNN as it would be with traditional image data. Probably the neural network most associated with time series prediction is the Long Short-Term Memory network (LSTM) [76], which is an example of Recurrent Neural Network. LSTM schema is presented in figure **??**. Long Short-Term Memory network adds native support for input data which are sequence of observations. It also understand that the data may become obsolete over time and that the newest information may be more relevant than the information learned some time ago. The presented architectures are examples of best known and established artificial neural networks, but the list do not end here. There are many more complex architectures but this is not the topic of this doctoral thesis.

,,

,,,,

3 Promotions

Forecasting the impact of promotions is a critical aspect of demand forecasting, as it helps companies to better understand the expected changes in customer behavior and demand patterns during promotional events. Accurate promotion forecasting enables companies to make informed decisions regarding product pricing, inventory management, and supply chain optimization. It also helps to avoid overstocking or stock shortages, which can negatively impact customer satisfaction and, ultimately, the bottom line. As a result, effective promotion forecasting is essential for maximizing the benefits of promotions while minimizing the associated risks and costs.

When discussing forecasting price reduction, we can distinguish two cases. First is connected mostly with eatable products or products that have an expiry date. In case of these products, when an expiry date is approaching, the price reduction is a necessity to encourage customers to buy the product and to minimize losses. Similar situation can be observed also in other industries, e.g. when sales of existing products are to be discontinued and units already produced need to be sold off.

Another case is planned promotions, which do not aim to sell out all units of a product in order to minimize losses. In these cases, price reduction has multiple business goals. This is strongly visible in case of *fast-moving consumer goods* (FMCG), e.g. food retailing. In large shop chains which sells everyday products, we can observe multiple promotions at the same time, which changes quickly, and they are replaced by other promotions. In most cases, always some kind of promotions is happening and on many products. Since there are numerous FMCG merchants on the market, it's important to maintain competitive. Offering products in promotion is one approach to achieve this. The amount of money spent on promotions in the FMCG sector each year — e.g. \$1 trillion in 2014, according to [43] — can demonstrate the significance of this practise. As a result, it's critical to predict how promotions may affect sales and plan accordingly.

In this chapter, I will focus on predicting the efficiency of planned promotions. I will base my research on real data from FMCG retail company.

Promotions are sometimes planned using judgmental forecasting or a simple baseline statistical forecast with a judgmental adjustment, as in [56]. This means that promotion planning is frequently done manually. However, as it was shown in chapter 1, studies have shown that relying solely on these types of forecasting methods can lead to bias. It might be a better idea to use more advanced methods that rely heavily on knowledge derived from historical data. Because of this, statistical and artificial intelligence methods can be used. Forecasting was traditionally done using statistical methods such as exponential smoothing [64], moving average, the Auto-regressive Integrated Moving Average (ARIMA) model and Seasonal Auto-Regressive Integrated Moving Average (SARIMA). In the field of sales forecasting, more complex methods were used and evaluated over time. A comparison of various linear and non-linear models for this task was performed in [40]. The neural network built on deseasonalised time series data produced the best results. The findings suggested that non-linear models should be heavily considered when modelling retail sales. Back-propagation neural network (BPNN) was another neural network algorithm used for forecasting retail sales for this task, according to [36]. In [16], evolutionary neural networks (ENN) were also considered. In this area, the use of the extreme learning machine (ELM) algorithm was also investigated, for example, in the papers [145], [175], and [172]. In addition, authors of [94] propose incorporating linguistic knowledge into the forecasting process using linear regression. An intriguing forecast technique was presented in the paper [161]. To forecast monthly car sales, the authors combined pre-purchase online search data with economic variables. Making sales projections for short shelf-life food products, often known as Fast-Moving Consumer Goods, is an important element of retail forecasting (FMCG). It is a much more difficult duty because the additional products, whose sales may be exaggerated, cannot be stored in the shop for an extended period of time. In the study [52], a radial basis function (RBF) neural network and a customized genetic algorithm were successfully employed to estimate fresh milk sales. The authors of [148] shown the benefits of using Machine Learning approaches in constructing demand forecasting models in the context of FMCG. The Autoregressive Distributed Lag model was used in the paper [78]. The authors of [6] proposed using a Dynamic Artificial Neural Network to forecast food sales for one of India's multiplexes. Different classifiers were examined in the paper [84], and a proposal for combining various forecasting models using neural networks was presented in order to improve results for forecasting warehouse demand. Regarding sales forecasting, decision and regression tree-based methods were also considered. In [150], a hybrid method of the k-means algorithm and the C4.5 algorithm (decision tree classifier) was demonstrated. A comparison of different Machine Learning Techniques to retail sales forecasting was conducted in the paper [101]. The authors concluded that boosting algorithms outperformed traditional regression algorithms. The GradientBoost algorithm produced the best results for them, and the XGBoost implementation was used to improve accuracy.

Forecasting sales during promotions is a difficult task, as stated in [56]. The authors of this paper pointed out that the promotional effect was typically estimated by combining simple statistical forecasting methods and adding judgmental adjustment, which could lead to errors. Sales promotion effectiveness research has been conducted, primarily in the marketing research area, and is described in practitioner literature. This issue was discussed in [25] and [176]. Authors of [43] proposed a new formula for the promotion optimization problem in the FMCG industry. Despite the fact that these works are about estimating the effectiveness of promotions, they all focus on domain knowledge and do not use machine learning techniques to do so. In the paper [158], multiple models for forecasting demand during promotional periods were tested. In order to predict sales in the presence of promotions, the paper [153] presented the use of PCA and pooled regression. Machine learning methods for direct marketing were compared and tested in the paper [44]. In paper [9] the authors demonstrated that simple statistical methods performed admirably on data with no promotions. However, more advanced methods had to be used during promotional periods. In this paper, regression trees were used to forecast grocery sales.

XGBoost [37] method was used in case of sales forecasting. In [157] authors studied and evaluated classical statistical approaches and machine learning methods. As a traditional method, they predicted daily sales of fruits and vegetables using seasonal ARIMA. LSTM and XGBoost were used as machine learning techniques. The results shown that the XGBoost algorithm produces more accurate results than the other two approaches. Another examples of successful use of XGBoost in case of sales forecasting can be found in [122], [47], and [69]. Variation on XGBoost method is also proposed in [90] and [167]. These are the examples of using XGBoost in case of sales forecasting. To the best of my knowledge, before my work on this topic, the tree boosting algorithm, especially the extreme gradient boosting (XGBoost) algorithm, has not been used to forecast the effect of promotions and to optimize the promotion itself.

3.1 Problem statement

The purpose of promotions is not only selling as much product as it is possible. Promotions should give the company bigger profit, and the aspect of increasing sales is important, but it is not the only one.

In article [73], for which I received the Professor Zdzislaw Pawlak Best Student Paper Award at the FedCSIS 2020 conference, we proposed six indicators in order to capture the effectiveness of promotion. These were the following indicators:

- Average number of sold units or kilograms each day (shortcut: Avg. Amount) – This indicator shows how many units or kilograms of the promoted product, on average, were sold during the promotion each day.
- Average number of receipts with the promoted product (shortcut: Avg. Nb. Receipts) – The indicator explains in how many baskets the promoted product appeared, on average, each day during the promotion. It can be treated as an indicator of how many customers bought the product each day.
- Average value of a basket containing the promoted product (shortcut: Avg. Basket) – This indicator says what an average value of a basket was where the promoted product appeared. Assuming that customers went for shopping with the will to buy the specific product in promotion, the indicator says how much money they spent in total. The higher the indicator, the more products were bought or the more expensive products were chosen.
- Average value of a basket containing the promoted product but disregarding the value of the promoted product (shortcut: Avg. Basket Without Item) – This indicator is very similar to the previous one. It shows what an average value of a basket was where the promoted product appeared, but the value of the promoted product was not taken into account. It means that this indicator is equal to 0 if the customer buys only the promoted product.
- Average number of unique products in the basket (shortcut: Avg. Nb. Unique Items) – It says how varied the basket is. The higher the value of the indicator, the better – it means that the customer not only bought a specific product, but also many others.

- Average number of the baskets (shortcut: Avg. Nb. Clients) – The indicator shows how many, on average, transactions were performed each day during the promotion. It does not matter if the customer bought a promoted product or not.

These indicators are *gain-type measures*, so it means that the higher the value the better. They may also be inversely correlated. One promotion can be described by all of these indicators. Using these indicators we can assess if promotion was satisfactory in all aspects or maybe only in case of one indicator. The retailers want to propose promotions that will increase sale of a product, but also will increase sale of other products that will be bought with the promoted product. Good promotions should also encourage new customers to visit a store.

Suggested indicators may be used to precisely describe the effect of historical promotions, but they can also be used to evaluate future promotions. It would be beneficial in case of promotions planning, because modelling the effect of combination of features on the result of future promotion, it is possible to find out if the promotion will have satisfactory effect.

Forecasting the promotion effect can be done separately for each product. With the history of promotions and their impacts, we can model the characteristics of the promotion for the specific product and anticipate what the effect will be in the future. However, because the number of previous promotions for many products is small, there are few examples for training a model. Another problem is how to predict the effect of promotion for a product that has never been in promotion or a product which was recently introduced. In order to cope with these issues, we proposed to group products and create forecasting models for each group of products. One approach could be based on domain knowledge. In this case, the expert could point out which products act similarly during promotions and products with similar promotion effect would be grouped. However, we did not have such knowledge and because of this we decided to create forecasting models for predefined groups of products.

To summarize: in [73] we proposed a new approach to forecast the effectiveness of promotions. For each of the 6 indicators and each group of products, a forecasting model was trained.

The research was carried out as part of the project co-financed by European Funds entitled 'Decision Support and Knowledge Management System for the Retail Trade Industry (SensAI)' (POIR.01.01.01-00-0871/17-00). The work was done in

store	product	start date	end date	conditional attributes	indicator
10	pears	2018-01-22	2018-01-25	...	123.56

Table 3.1: Example of record describing promotion.

cooperation with the company 3Soft S.A. that provides support to, among others, one of the largest discount store chains in Poland and Europe.

3.2 Datasets

In research on predicting the efficiency of promotion, we used datasets from a large grocery retail company with more than 500 stores. Because the data is sensitive to the client, it could not be shared. The data from three groups were investigated: fruits, vegetables and dairy products. In our studies, only regular promotions were taken into account. These were the promotions that did not happen before or during holidays and no additional condition had to be met for the discount to apply (e.g. no special combination of products had to be purchased together, and the discount did not apply only if multiple items were purchased). Also, promotions due to approaching expiry date were not included in research. Only regular promotions were used because they were a majority of all promotions.

We were working on data from 2015 to 2018. Data from 2015 and part of 2016 was not complete, and there was visible smaller number of promotions from this period. We prepared data in this way that one record described one promotion in one store. Therefore, for example, if there would be a promotion on pears in the store with ID 10 from 2018-01-22 to 2018-01-25, the record would look like in table 3.1.

In our paper, we proposed conditional attributes that describe promotion in a store. We created features:

- connected with price
  - price of a product,
  - change of a price;
- connected with the time and duration of the promotion
  - number of days of the promotion,
  - weekday of the first day of the promotion,

  - other attributes created based on a date of the first day of promotion: year number, month number, day number, week number, number of a day in the year, and the season;

- describing the advertisement media (promotion channels)
  - logical value describing if promotion was advertised in TV,
  - logical value describing if promotion was advertised in radio,
  - logical value describing if promotion was advertised in Internet,
  - logical value describing if promotion was advertised in a different way,
  - new variables describing combinations of the promotion channels – for each combination of basic attributes describing promotions, new attributes were created as a result of binary operations AND, OR and XOR (only when the combination consisted of 2 elements);
- describing the store and its surroundings
  - number of inhabitants within 1 km,
  - number of inhabitants per 1 square km,
  - number of inhabitants within a 5-minute driving range,
  - unemployment rate,
  - number of cars per 1,000 inhabitants,
  - average monthly salary,
  - tourism ratio, etc.
- describing the impact of other promotions
  - number of all promotions in a store,
  - number of all promotions that were advertised on TV, radio or internet,
  - number of all promotions that were advertised on TV, radio, internet or in a different way;

Attributes describing price are self-explanatory. Attributes describing time of promotion, so duration of promotion and date of first day of promotion, were added in order to catch the impact of time of promotion on its effectiveness. We also assumed that promotions in similar stores (for example in small villages or in big cities) can have similar characteristics. Because of this, extended features describing the shop and its surroundings were added to datasets. Another set of attributes were added in order to find the relationship between advertisement channels and the effectiveness of promotion. Unfortunately, we were only provided with binary information if the advertisement has taken place. We did not have information on the timing of the advertising and its frequency, and the costs incurred to pay for the transmission of the advertising. For this reason, these features may not provide complete information. We also proposed to add features describing combination of promotion channels in order to somehow capture the scale of advertisement. Exemplary new features: promotion was on TV or on the radio. (OR operation), promotion was on TV or on the radio or on the Internet (OR operation), promotion was on the TV and on the radio (AND operation), promotion was either on the Internet or on the radio (XOR operation). Because multiple promotions in grocery stores take place at the same time, we also introduced attributes describing the number of other promotions. We assumed that the more promotions in the shop, the more customer interest.

Another step in preparing datasets was finding matching periods without promotions in order to capture the characteristics of a product. The matching period had to meet the following conditions:

- It considered the same product as the promotion.
- It considered the same store.
- It had to last as many days as the considered promotion.
- It had to start on the same weekday as the promotion.
- The considered product was not in promotion on any given day.
- The period without promotion could occur maximum 4 weeks and minimum 1 week before the promotion.

The schema of finding matching periods without promotions is shown in figure 3.1. Next, we decided to perform standardization of two indicators:

- Average number of sold units or kilograms each day and
- Average number of receipts with the promoted product.

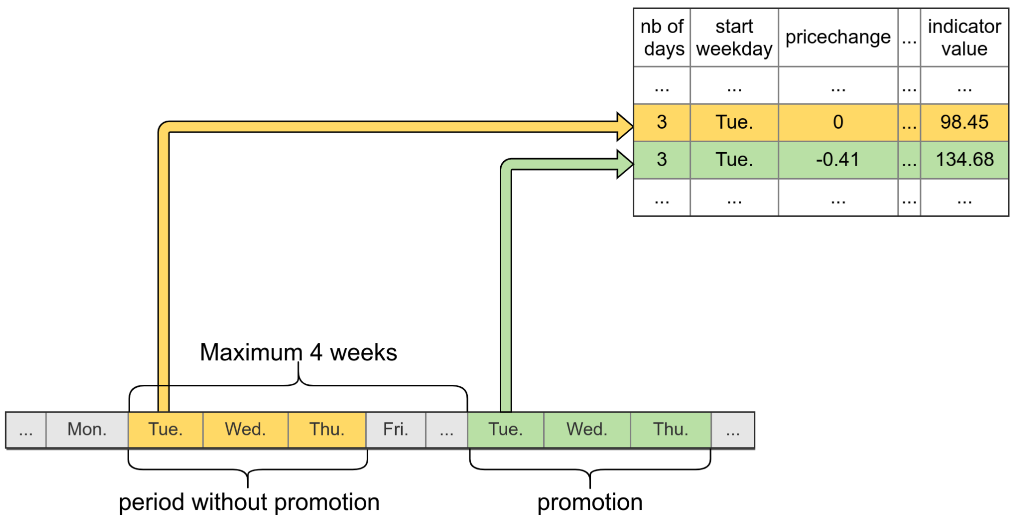


Figure 3.1: Finding matching record without promotion

Standardization was performed for each store and each product separately. Z-score standardization method was used. Such data modification was performed because these two indicators are correlated with sale characteristic of the product. This means that two products that naturally differ in sales volume, e.g. due to a large price difference, will have different sales characteristics, but this does not mean that the same promotion, will not affect them equally. With standardization, we can remove the effect of natural sales volume on the concerned indicators values concerned.

3.3 Modelling effectiveness of promotions

In order to model effectiveness of promotions, we created a predictive model for each group of products and each indicator. Thanks to this, for each planned promotion, we can evaluate the predicted effect of promotion.

We used XGBoost (eXtreme Gradient Boosting) [37] algorithm and its implementation from the R package `xgboost` [38] for training forecasting models. The algorithm was chosen because of its proven effectiveness on tabular sets [37]. Additionally, the paper [101] showed that this algorithm has given the best results for forecasting the sale in retail stores, so we assumed that it will give good results also for the problem of forecasting the promotion effect in retail sector.

In the experiments, data from 2015 to 2017 was used as training dataset. We included records describing promotions and matching periods without promotions. Data from 2018 was used as test dataset and in this set only promotions were included.

Cross-validation was not performed because, even though the dataset was not typical time series, the data were still linked to the chronology of occurrence. The same conditional attributes were used for each indicator and the value of the indicator was a decision attribute (for indicators Average number of sold units or kilograms each day and Average number of receipts with the promoted product values after standardization were used as decision attribute).

In the process of evaluating the results, we used the following error metrics:

- *Mean Absolute Error (MAE)*:

$$MAE = \frac{\sum_{i=1}^{n} |F_i - A_i|}{n}$$

- *Root Mean Square Error (RMSE)*:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} (F_i - A_i)^2}{n}}$$

- *Mean Absolute Percentage Error (MAPE)*:

$$MAPE = \frac{1}{n} \sum_{i=1}^{n} \left| \frac{A_i - F_i}{A_i} \right|$$

- *Weighted Mean Absolute Percentage Error (WMAPE)*:

$$WMAPE = \frac{\sum_{i=1}^{n} \mid A_i - F_i \mid}{\sum_{i=1}^{n} A_i}$$

where A_i is the actual value and F_i is the forecast value.

The mean absolute percentage error (MAPE) is a very straightforward and simple to understand error metric, but it is only meaningful when the values are large. If the actual value is near to zero, the MAPE value approaches infinity and produces uninterpreted results. To get around these drawbacks, a similar metric called WMAPE was adopted that works better with smaller values. It is commonly used in the retail industry. MAPE and WMAPE measures are mainly used because of their interpretability and understandability for non-data analysts. However, further studies connected with data analysis can still use measures such as RMSE or MAE.

We also included a process of XGBoost hyperparameters optimization using *grid search* method. The optimization was performed using the validation set that was extracted from the training data set. Six hyperparameters were optimized:

- *nrounds* – maximum number of boosting iterations; range: $[1, \infty)$. We tested values: 1, 21, 41 ..., 181, 201.
- *base_score* – the initial prediction score of all instances; range: $(-\infty, \infty)$. Tested values depended on indicator values and were calculated as 11 quantiles from indicator values with the following probabilities: 0.0, 0.1, 0.2, ..., 0.9, 1.0.
- *eta* – boosting learning rate; range: $[0, 1]$. We tested values 0.0, 0.1, 0.2, ..., 0.9, 1.0.
- *gamma* – minimum loss reduction required to make a further partition on a leaf node of the tree; range: $[0, \infty)$. We tested values 0, 1, 2, ..., 9, 10.
- *max_depth* – maximum depth of a tree; range: $[1, \infty)$. We tested following values: 1, 4, 7, 10, 13.
- *subsample* – subsample ratio of the training instance; range: $(0, 1]$. Our tested values were: 0.0001, 0.1001, 0.2001, ..., 0.9001.

The flowchart of the optimization process is shown in figure 3.2.

The results of models efficiency, calculated for the test data sets after hyperparameters optimization, is shown in table 3.2.

The optimization was carried out based on the RMSE measure. The improvement of this metric was observed for every examined case. The table 3.3 shows the exact results.

In the article [72], for which I am a first co-author, we investigated 4 from 6 proposed indicators in order to present and compare the abilities of forecasting promotion effect using the gradient boosting method with Deep Learning approach. Chosen indicators are:

- Average value of a basket containing the promoted product
- Average value of a basket containing the promoted product but disregarding the value of the promoted product
- Average number of unique products in the basket
- Average number of the baskets

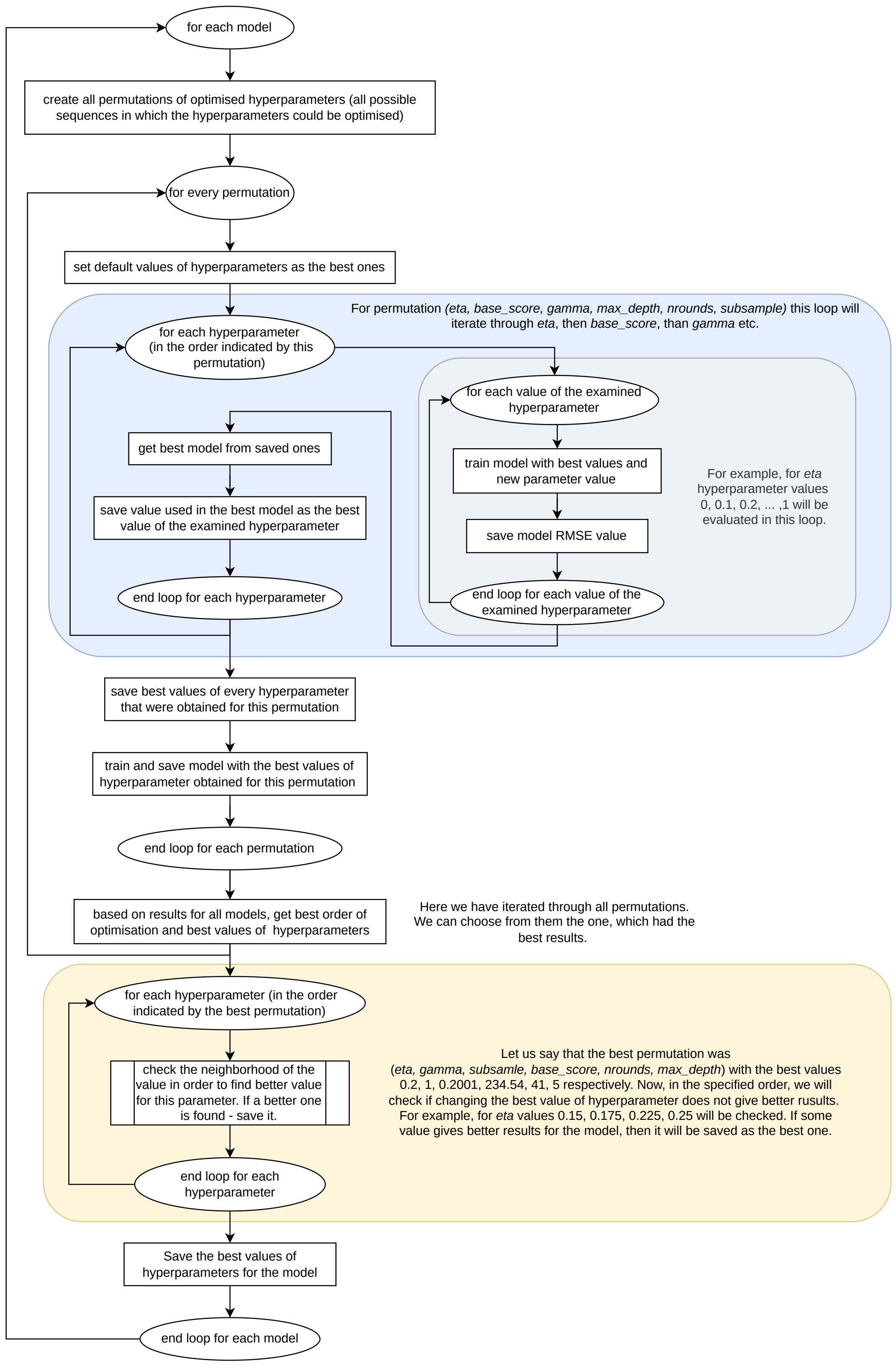

for each model
create all permutations of optimised hyperparameters (all possible sequences in which the hyperparameters could be optimised)
for every permutation
set default values of hyperparameters as the best ones
For permutation (eta, base_score, gamma, max_depth, nrounds, subsample) this loop will iterate through eta, then base_score, than gamma etc.
for each hyperparameter (in the order indicated by this permutation)
for each value of the examined hyperparameter
train model with best values and new parameter value
save model RMSE value
end loop for each value of the examined hyperparameter
For example, for eta hyperparameter values 0, 0.1, 0.2, ... ,1 will be evaluated in this loop.
get best model from saved ones
save value used in the best model as the best value of the examined hyperparameter
end loop for each hyperparameter
save best values of every hyperparameter that were obtained for this permutation
train and save model with the best values of hyperparameter obtained for this permutation
end loop for each permutation
based on results for all models, get best order of optimisation and best values of hyperparameters
Here we have iterated through all permutations. We can choose from them the one, which had the best results.
for each hyperparameter (in the order indicated by the best permutation)
check the neighborhood of the value in order to find better value for this parameter. If a better one is found - save it.
Let us say that the best permutation was (eta, gamma, subsamle, base_score, nrounds, max_depth) with the best values 0.2, 1, 0.2001, 234.54, 41, 5 respectively. Now, in the specified order, we will check if changing the best value of hyperparameter does not give better rusults. For example, for eta values 0.15, 0.175, 0.225, 0.25 will be checked. If some value gives better results for the model, then it will be saved as the best one.
end loop for each hyperparameter
Save the best values of hyperparameters for the model
end loop for each model

Figure 3.2: Flowchart of the XGBoost hyperparameter optimization process.

Table 3.2: Results of models effectiveness after hyperparameters optimization

category	indicator	MAE	RMSE	MAPE	WMAPE
dairy products	AVG. AMOUNT	12.31	18.71	0.53	0.38
dairy products	AVG. NB. RECEIPTS	5.75	8.08	0.45	0.33
dairy products	AVG. BASKET	13.93	20.14	0.17	0.17
dairy products	AVG. BASKET WITHOUT ITEM	14.26	20.32	0.19	0.18
dairy products	AVG. NB. UNIQUE ITEMS	2.04	2.72	0.14	0.13
dairy products	AVG. NB. CLIENTS	129.75	177.15	0.08	0.08
fruits	AVG. AMOUNT	39.72	74.62	1.11	0.45
fruits	AVG. NB. RECEIPTS	24.87	39.39	0.85	0.35
fruits	AVG. BASKET	15.29	22.44	0.16	0.16
fruits	AVG. BASKET WITHOUT ITEM	14.73	21.78	0.17	0.17
fruits	AVG. NB. UNIQUE ITEMS	1.84	2.60	0.12	0.11
fruits	AVG. NB. CLIENTS	125.15	164.49	0.07	0.07
vegetables	AVG. AMOUNT	22.97	42.42	0.47	0.33
vegetables	AVG. NB. RECEIPTS	19.52	34.78	0.41	0.31
vegetables	AVG. BASKET	14.39	21.56	0.15	0.15
vegetables	AVG. BASKET WITHOUT ITEM	14.63	21.65	0.16	0.16
vegetables	AVG. NB. UNIQUE ITEMS	1.89	2.71	0.11	0.11
vegetables	AVG. NB. CLIENTS	135.57	178.51	0.08	0.08

Table 3.3: RMSE improvement after hyperparameter optimization. $RMSE_{diff}$ is a difference of RMSE before optimization and RMSE after optimization.

	$RMSE_{diff}$		
indicator	dairy products	fruits	vegetables
AVG. AMOUNT	0.78	9.72	2.47
AVG. NB. RECEIPTS	0.07	5.83	2.71
AVG. BASKET	1.38	4.19	4.74
AVG. BASKET WITHOUT ITEM	0.65	3.31	3.88
AVG. NB. UNIQUE ITEMS	0.13	0.56	0.53
AVG. NB. CLIENTS	70.39	14.13	50.49

Table 3.4: Values of hyperparameters used in optimization process of Deep Learning models.

layers and hyperparameter	tested values
First Dense layer – sizes of the output space	64, 256, 512, 1024
First Dropout layer – fractions of the input units to drop	from 0 to 1
Second Dense layer – sizes of the output space	64, 256, 512, 1024
Second Activation layer – activation function	*relu*, *sigmoid*
Second Dropout layer – fractions of the input units to drop	from 0 to 1
conditional additional layers – Dense layer – sizes of the output space	10, 50, 100
Conditional additional layers – Dropout layer – sizes of the output space	from 0 to 1
Optimizer	*rmsprop*, *adam*

The indicators were investigated because for them Z-score standardization didn't have to be performed, making the comparison of the selected methods more explicit.

In these experiments also three groups of products were investigated, and for each indicator and each category of product we created separate forecasting models. The experiments were conducted with train and test data sets. The train data sets had records with promotions and matching records without promotions from 2015 to 2017. The test data sets consisted only of promotions from 2018. We investigated XGBoost, Sequential Deep Learning Models and Deep Learning Models from H2O. The first one was already described. In the experiments with XGBoost also grid search method was used for hyperparameters optimization.

Sequential Deep Learning models were created using Python language with the usage of the `Keras` library [39]. `Hyperas` library [129] was used for hyperparameters optimization. This library aids in the creation of templates for Keras hyperparameter search spaces in order to identify the best combination of their values. The table 3.4 shows values that were tested by `Hyperas` library. In this process, the validation data sets, extracted from the training data sets, were also used.

Next, we tested Deep Learning models from H2O automodel module (AutoML). We used for this R Interface for H2O – `h2o` [4]. AutoML in H2O can be used to discover the best model with the best combination of hyperparameters for a given task.

The results of these comparisions are presented in Figure 3.3 and in Figure 3.4. The Figure 3.3 shows the values of RMSE and the Figure 3.4 shows the values of MAE.

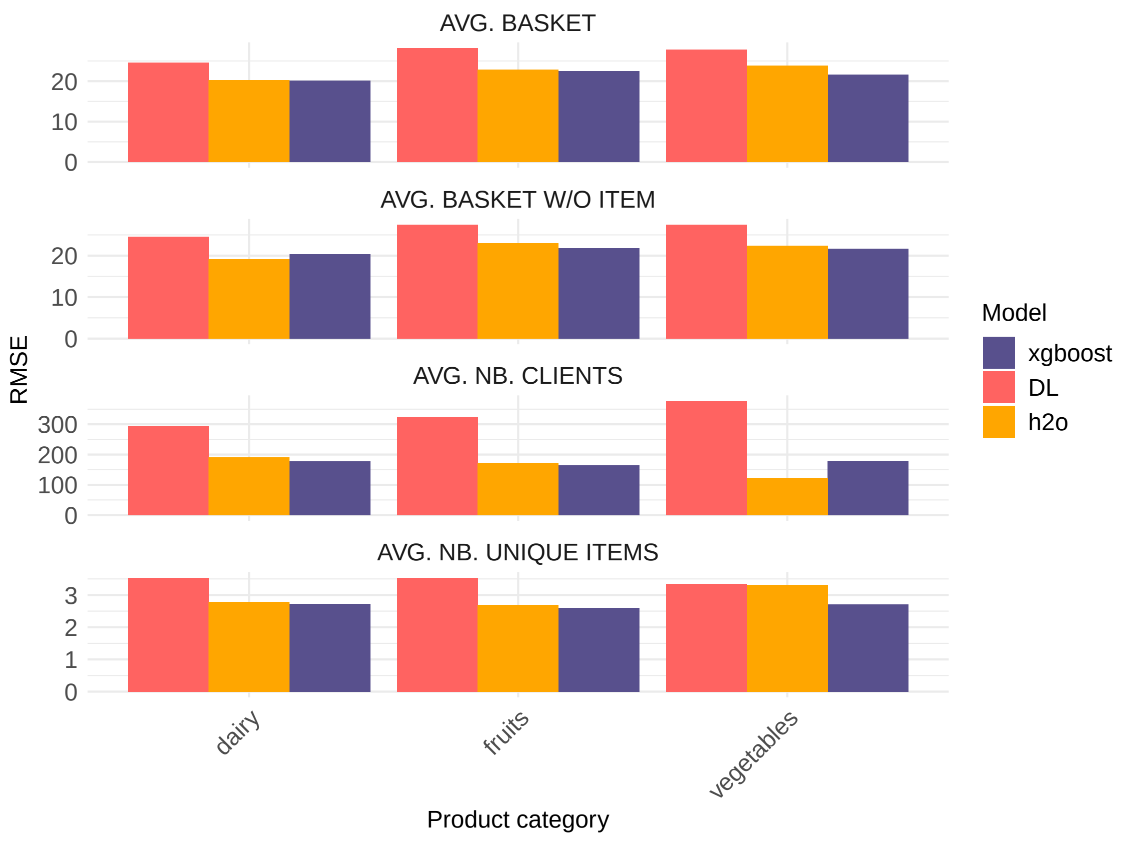


Figure 3.3: RMSE of results obtained in the experiments for the XGBoost models (*xgboost*), for Deep Learning models made with Keras library (*DL*) and H2O Deep Learning models obtained from AutoML (*h2o*).

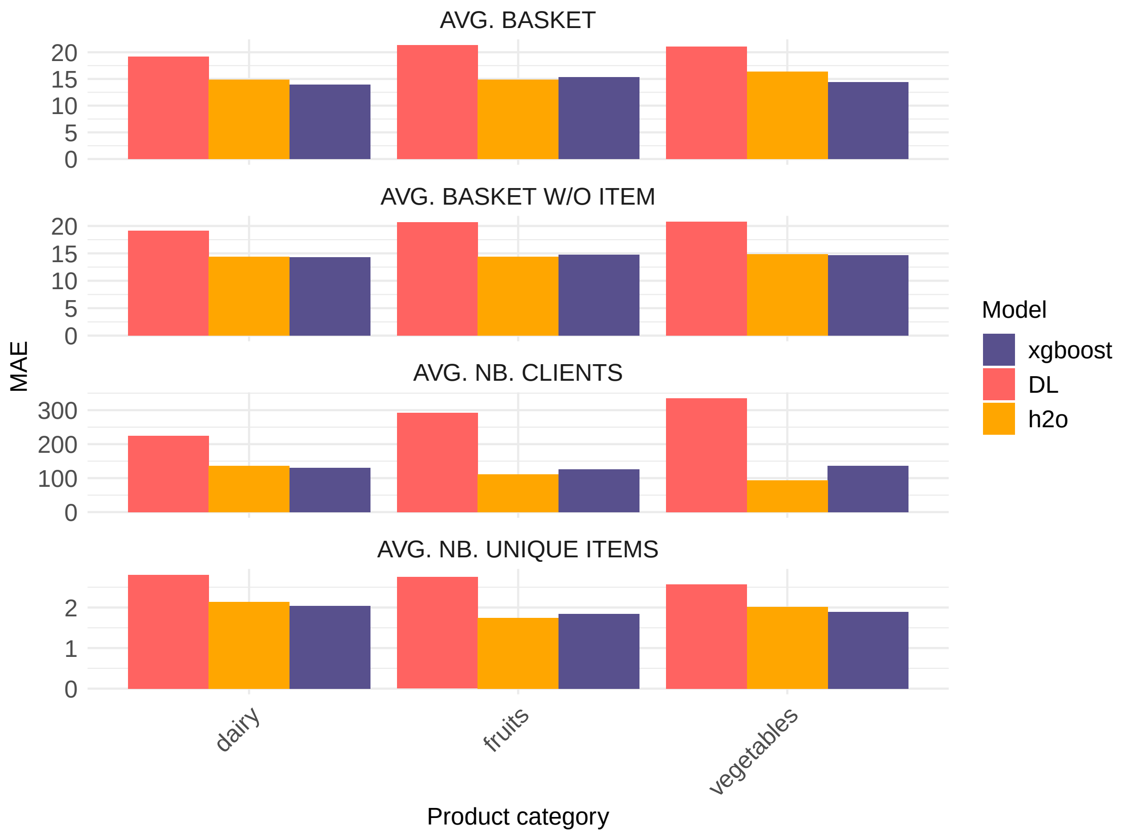


Figure 3.4: MAE of results obtained in the experiments for the XGBoost models (*xgboost*), for Deep Learning models made with Keras library (*DL*) and H2O Deep Learning models obtained from AutoML (*h2o*).

From the results presented in Figures 3.3 and 3.4 we can see that in most cases the approach using XGBoost was the best. Taking into account the measure that was optimized – RMSE – we obtained the best results for XGBoost in 10 out of 12 cases. In the other cases, Deep Learning models from a H2O automodel were the best. Deep Learning models made in Keras and optimized by `hyperpot` library had visible lower results.

From these results, we can conclude that the initially chosen algorithm for the task of predicting values of promotion effectiveness indicators – XGBoost algorithm – gave generally better results than Deep Learning approaches. It was even better from H2O automodel, despite using custom grid search procedure.

Another research connected with forecasting promotion effectiveness were presented in article [24] in which I am a co-author. In this research, we focused on Avg. Amount and Avg. Basket indicators and we considered products separately (not as a part of group of products). It means that the forecasting models were created separately for each product and each indicator. We compared algorithms from 4 well-known families: tree-based (*Random Forest*), distance-based (*kNN*), neural networks and linear model. Then, the most promising family (with the best results) was further evaluated. Thanks to this approach, we didn't have to conduct tests for a wide range of methodologies, but we only focused on the most promising ones. Because of this, we extended the tree-based family approach by *Gradient Boosted Trees* algorithm, because the *Random Forest* method gave the best results from initially 4 tested algorithms. Finally, we considered the following algorithms:

- *Random Forest* (RF) [31] – decision tree based ensemble, where the ensemble members collectively vote for the final prediction
- *Gradient Boosted Trees* (GBT) [37] – boosting based approach, where the decision trees are constructed one by one in order to minimize some cost function. The implementation of this algorithm is e.g. XGBoost.
- *kNN* [99] – k-nearest neighbours
- *Generalized Linear Models* (GLM) [115] – linear model with automatic parameters running model
- Neural networks (*MLP*) [98] - an MLP neural network

In this process also optimization of hyperparameters was performed with *grid search* approach. The performance of models was measured using two different metrics:

- Root Mean Square Error (RMSE)
- Correlation

$$R = \frac{\sum_{i=1}^{n} (A_i - mean(A))\,(F_i - mean(F))}{std(A)\,std(F)}$$

where n is the number of samples in the data, A is the actual value and F is the forecast value.

Correlation (R) is a statistical measure that describes the relationship between two variables. It has a value between -1 and 1, where a perfect negative correlation is represented by a value of -1, a perfect positive correlation by a value of 1, and no correlation by a value of 0. A positive correlation indicates that the two variables tend to rise together as one rises. When two variables are negatively correlated, one variable tends to decrease as the other increases.

Together, these two measurements help us grasp the importance of error. In these tests, various products have various characteristics – some products are measured in kilograms and some are sold by the piece, so for example the same value of RMSE for different products can have completely different meaning. On the other hand, correlation measure has fixed range which made it easy to interpret results for different products, but it ignores the bias.

Results of RMSE for each product and each algorithm are presented in Figure 3.5 and 3.6. The obtained results indicate that out of the evaluated models, *Random Forest* and *Neural networks* usually are the best. We used the Wilcoxon signed-rank test to compare results of these two algorithms. The Wilcoxon signed-rank test is a non-parametric test used to compare two related groups when the data are not normally distributed. The null hypothesis for the Wilcoxon signed-rank test is that there is no difference between the medians of the two related groups. If the p-value of the test is less than the significance level (usually 0.05), then we can reject the null hypothesis and conclude that there is a significant difference between the two algorithms. For AVG. AMOUNT indicator the p-value is 0.109 so it means that there aren't significant difference for these two algorithms. However, p-value of Wilcoxon test for RF and kNN, RF and GLM, RF and GBT are smaller than 0.05 so we can reject null hypotesis of the test. On the plot 3.6 differences are less visible, however here also Wilcoxon test was calculated and for RF paired with each other algorithm always p-value was less than 0.05. We can also observe that the performance of *Generalized Linear Models* in comparison to the best models is lower. It indicates that the relation between the input variables and the outputs is nonlinear.

We also analysed the values of the correlation coefficient. It is relatively high – for Avg. Amount on average it is $R \simeq 0.93$, and for Avg. Basket it is $\bar{R} \simeq 0.8$. However, for some products this metrics drops even to 0.64. We have investigated the concerning large variance of results, and it turns out that one of the factors which influence forecasting accuracy is the size of the dataset. As shown in Figure 3.7 the correlation R correlates with the size of the data, and the correlation coefficient is $corr(datasetSize, R) = 0.62$. This confirms our original assumption that for selected products, for which we have a lot of data on historical promotions, predicting the effect of a promotion based only on the data of the product in question may be possible. However, for many products there may be too little of this data, in which case it may be useful to forecast the effect of the promotion by taking into account historical data on multiple products.

3.4 Practical applications of predictive models

In articles [73] and [72] we discussed a practical site of presented research. We highlighted that models, which predict the promotion indicator values and were trained for a group of products, can help in the process of planning future promotions. Using them, the person in charge of planning promotions can check if a proposed combination of characteristics would give satisfactory results of the evaluated indicators. In the articles, we discussed also the utility of analysing the feature importance and break-down plots for created models.

In subsequent research, I have further explored the usefulness of the models created.

3.4.1 Price sensitivity simulator

In the first iteration of the study, the aim was to use previously prepared data and a predictive model to create a prototype solution for modelling the impact of promotions on sales and profit. The prototype was prepared to simulate the value for Avg. Amount indicator, but in the same way other indicators could be simulated. The value of the indicator was scaled within a given product and a given shop, and

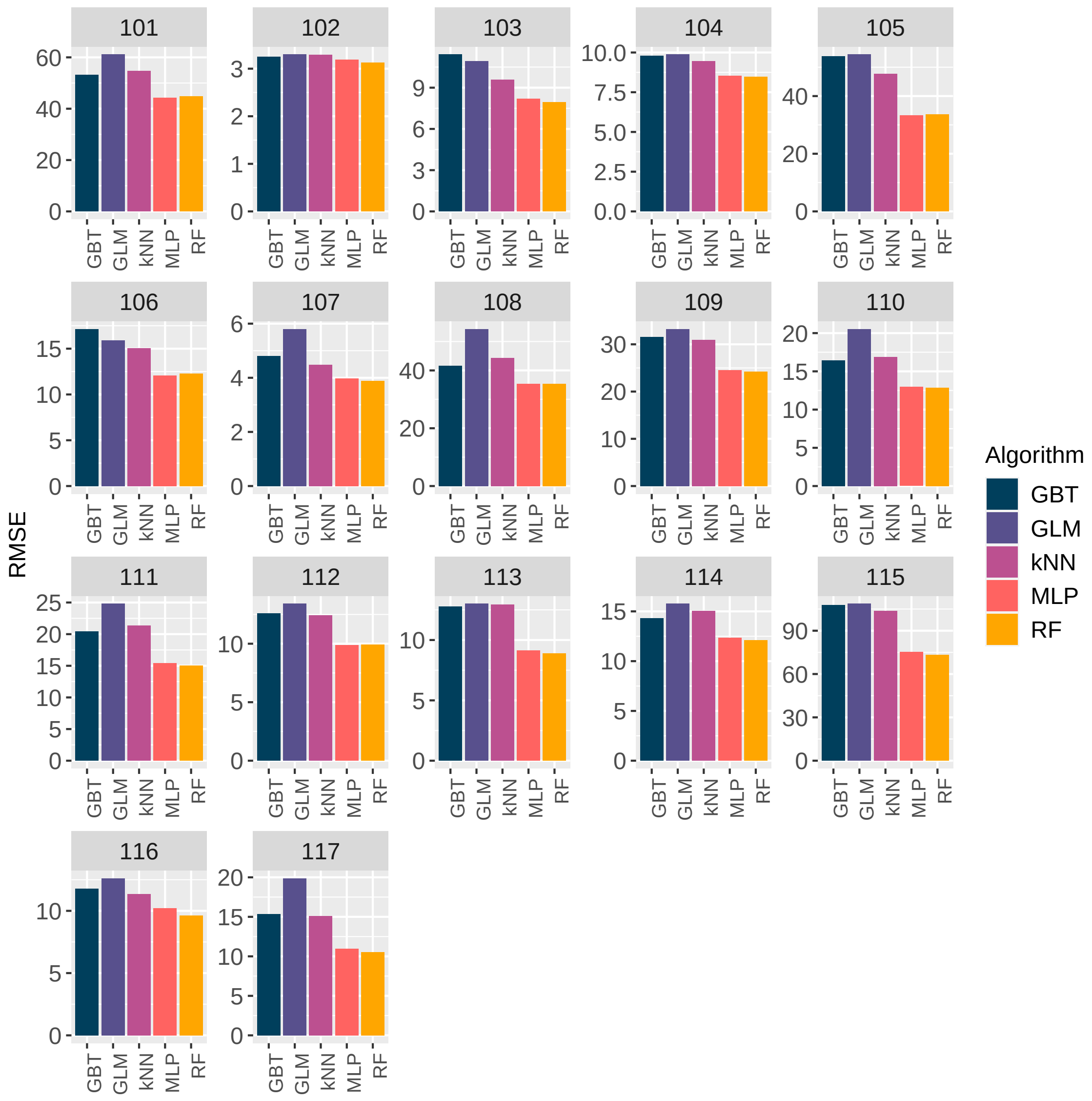


Figure 3.5: Prediction results for Avg. Amount for 17 products and 5 prediction models.

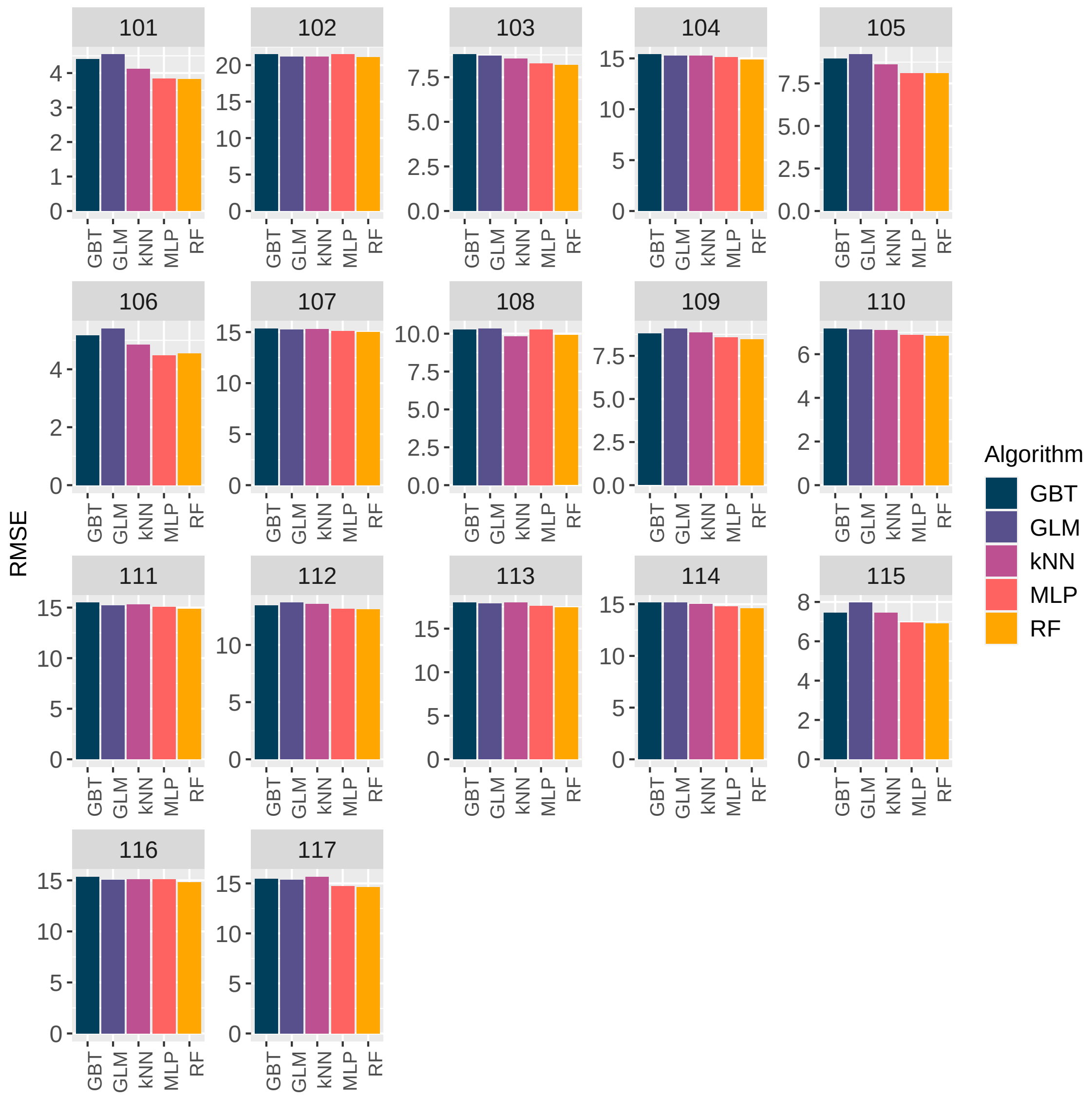


Figure 3.6: Prediction results for Avg. Basket for 17 products and 5 prediction models.

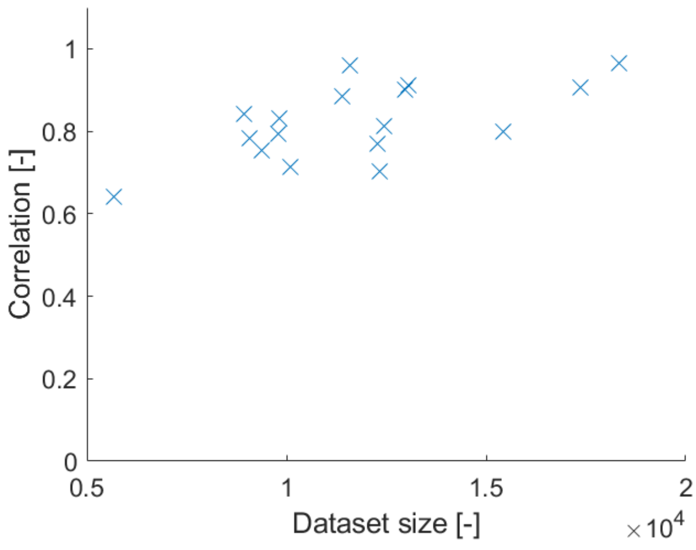


Figure 3.7: Relation between dataset size and R performance measure.

it was a predicted value. Then, it was inverted to the actual value before returning it to the user.

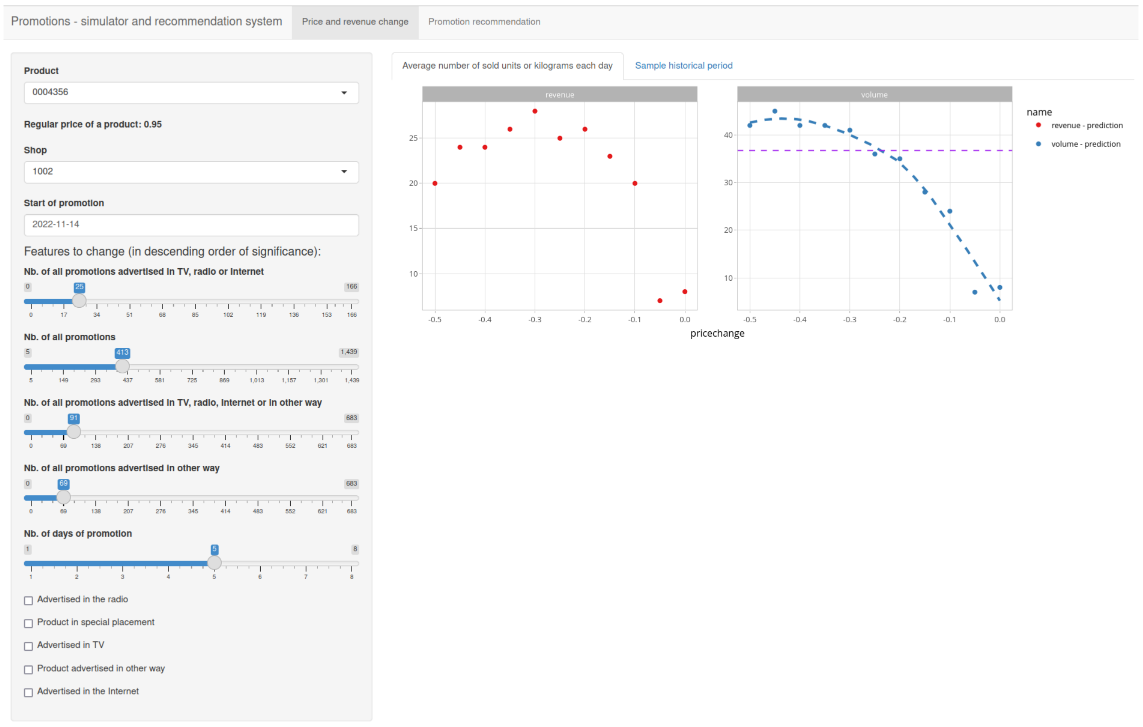
At the input, the user was to indicate for which product they wanted to create promotion. They would then decide in which shop the simulated promotion would be introduced and on which day it would start. Next, the user was to set other parameters that are configurable when planning the promotion, namely the number of other promotions taking place in the shop at the same time, the duration of the promotion, in which media the promotion would be advertised and whether the product would be specially arranged in the shop. The user using the simulator was to be able to change the promotion settings in real time and check the effect of the given changes.

At the output, the user was to receive information on how the sales of the product change for different price changes and how the profit changes (where profit is calculated as the product price multiplied by the sales forecast).

Interface

The work resulted in the development of a prototype application that is a price sensitivity simulator. The interface of a prototype is visible in Figure 3.8.

On the left-hand side of the app is the user configuration panel. In this section, the user selects which product and which shop the promotion applies to. They also


Promotions - simulator and recommendation system | Price and revenue change | Promotion recommendation

Product
0004356

Regular price of a product: 0.95

Shop
1002

Start of promotion
2022-11-14

Features to change (in descending order of significance):

Nb. of all promotions advertised in TV, radio or Internet: 25 (0–166)

Nb. of all promotions: 413 (5–1,439)

Nb. of all promotions advertised in TV, radio, Internet or in other way: 91 (0–683)

Nb. of all promotions advertised in other way: 69 (0–683)

Nb. of days of promotion: 5 (1–8)

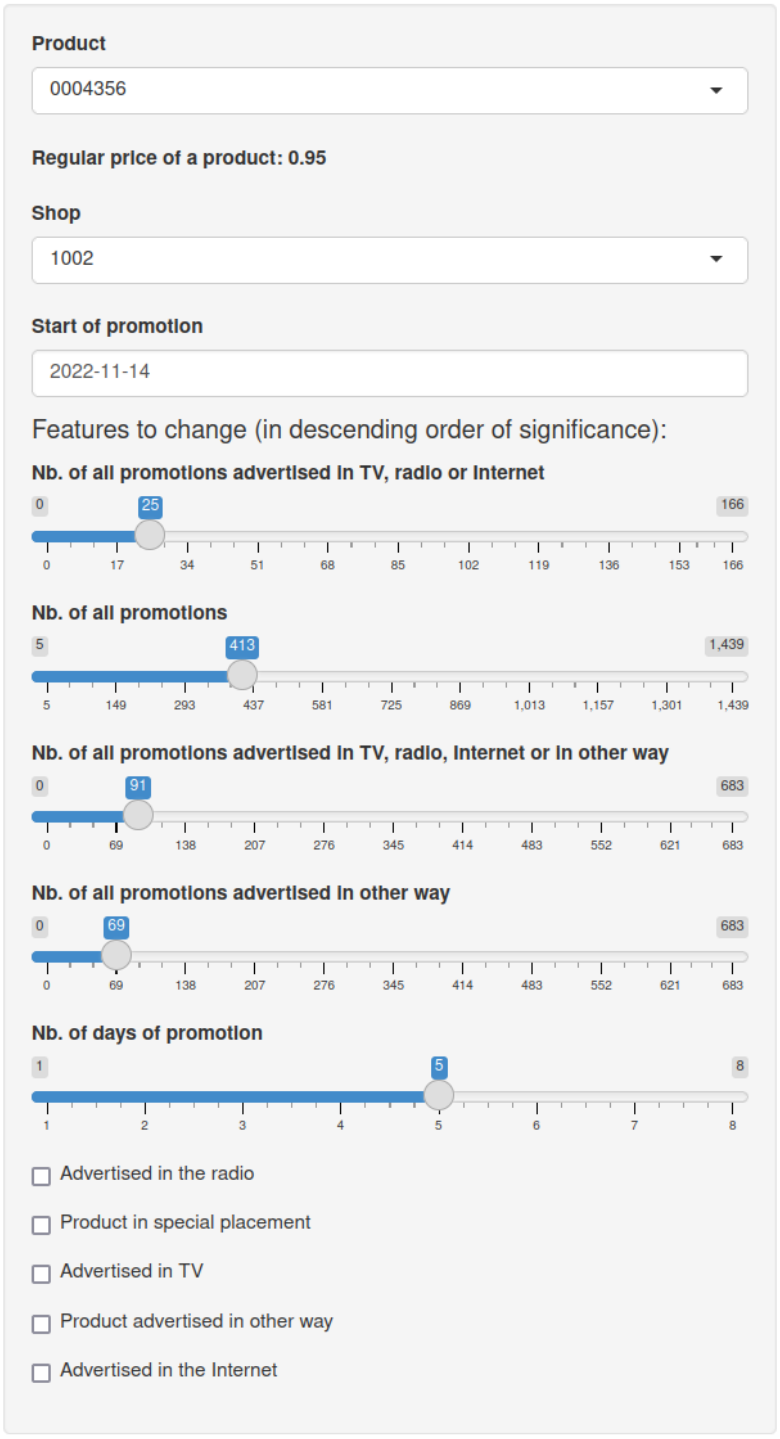
☐ Advertised in the radio
☐ Product in special placement
☐ Advertised in TV
☐ Product advertised in other way
☐ Advertised in the Internet

Figure 3.8: Interface of price sensitivity simulator.

set the start date of the promotion and other parameters describing the promotion. Under the product name, the regular price of the product is displayed. This price is taken from historical data. The configuration panel looks as in Figure 3.9. The values of the numerical variables are selected using a slider. The range of values is determined by historical data. Values for binary variables are selected using a checkbox. Default values for numeric and binary variables are also set based on historical data – these are rounded average values from the training set for the variable. The variables (sliders and checkboxes) are set in order from most significant to least significant in the predictive model.

There are two tabs on the right-hand side of the application. The first tab shows graphs which present sales and profit per product for different price changes. The price change was set between 0 and -0.5 with a step of 0.05. The price change is calculated as $\frac{promotional\ price}{regular\ price} - 1$. The graphs are visible in Figure 3.10.

The graph on the right (blue curve) shows how, for a given promotion setting, the model forecasts average daily sales over the duration of the promotion. The X-axis indicates the change in price (from -0.5 to 0) and the Y-axis is the projected average daily number of units sold. It can be seen that the graph with a larger reduction



Product

0004356

Regular price of a product: 0.95

Shop

1002

Start of promotion

2022-11-14

Features to change (in descending order of significance):

Nb. of all promotions advertised in TV, radio or Internet

0 — 25 — 166

Nb. of all promotions

5 — 413 — 1,439

Nb. of all promotions advertised in TV, radio, Internet or in other way

0 — 91 — 683

Nb. of all promotions advertised in other way

0 — 69 — 683

Nb. of days of promotion

1 — 5 — 8

☐ Advertised in the radio

☐ Product in special placement

☐ Advertised in TV

☐ Product advertised in other way

☐ Advertised in the Internet

Figure 3.9: Configuration panel of an interface of price sensitivity simulator.

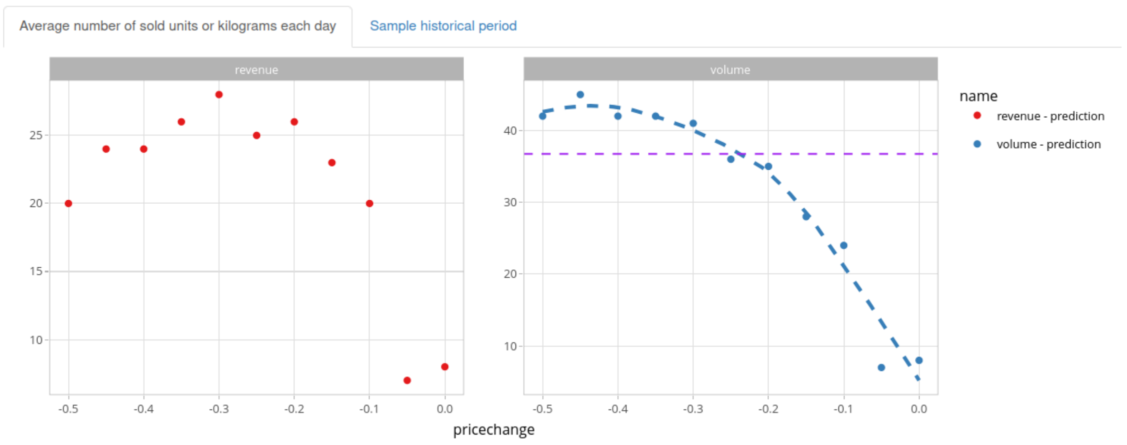
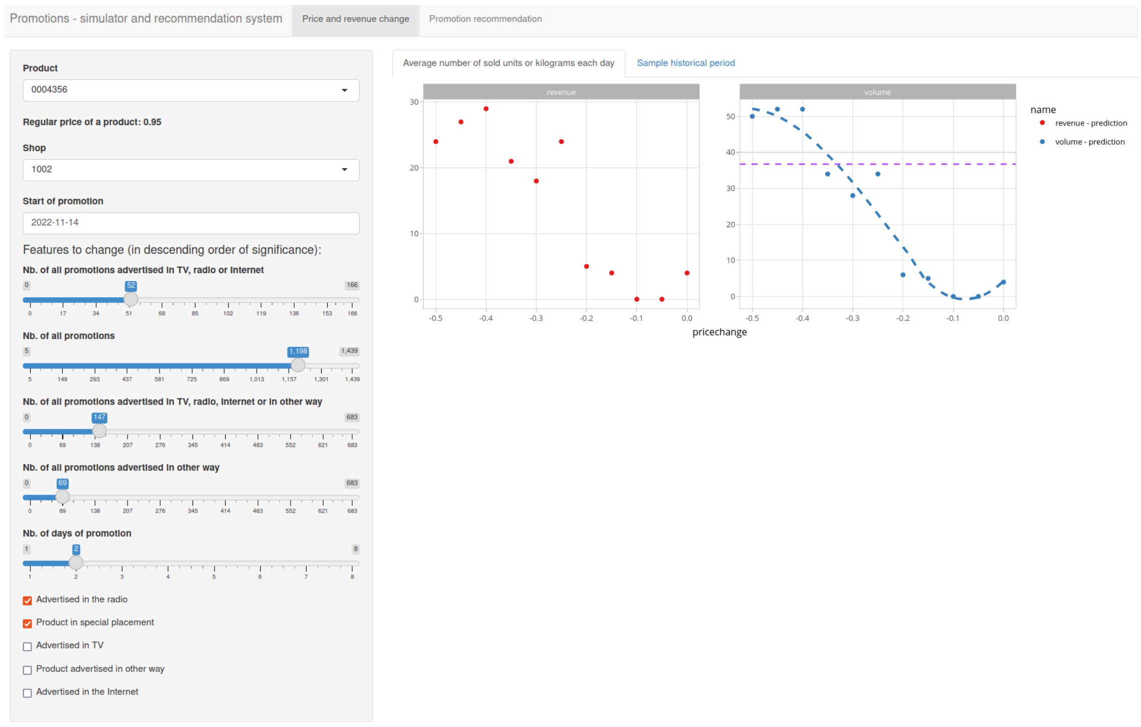


Figure 3.10: Plots for revenue and volume predictions.

(smaller value on the X-axis) indicates more sales. The dashed purple line indicates the 99th percentile of the average sales of a given product in a given shop. This gives us an indication of the observed maximum average sales of the product in the shop.

The graph on the left shows the average daily profit on sales of the product after taking into account the price change. The new price is calculated as (regular price)*(price change + 1). The profit is calculated as (new price)*(projected average daily sales of the product). On the X-axis is marked the price change (from -0.5 to 0) and on the Y-axis is the projected daily average profit on product sales. It can be seen that although the value of sales increases with a larger reduction (blue graph), in terms of profit, the largest reduction is not necessarily the most beneficial. From the graph above, it can be seen that, according to the forecasting model, the highest profit will be obtained for a reduction of 30% (value -0.3 on the X axis). The given graphs change as the given promotion parameters change what is presented in Figure 3.11.

The second tab of the module displays the description of the promotion that is used by the predictive model. It is visible in Figure 3.12. All attributes, whose values are created based on values from the configurator, are already displayed as input to the model. This tab describes the base example, which will be described in more detail in the subsection describing the application logic.



Product: 0004356

Regular price of a product: 0.95

Shop: 1002

Start of promotion: 2022-11-14

Features to change (in descending order of significance):

Nb. of all promotions advertised in TV, radio or Internet: 52 (0–166)

Nb. of all promotions: 1,198 (5–1,439)

Nb. of all promotions advertised in TV, radio, Internet or in other way: 147 (0–683)

Nb. of all promotions advertised in other way: 69 (0–683)

Nb. of days of promotion: 2 (1–8)

[✓] Advertised in the radio

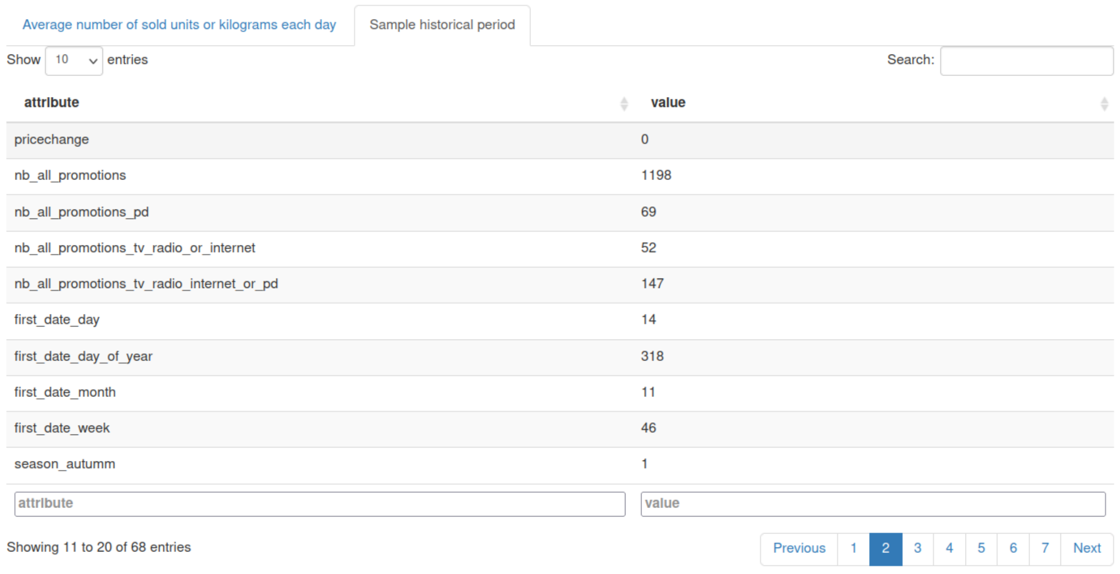
[✓] Product in special placement

[ ] Advertised in TV

[ ] Product advertised in other way

[ ] Advertised in the Internet

Figure 3.11: Change of plots for revenue and volume predictions after changing values in a configuration panel.



attribute	value
pricechange	0
nb_all_promotions	1198
nb_all_promotions_pd	69
nb_all_promotions_tv_radio_or_internet	52
nb_all_promotions_tv_radio_internet_or_pd	147
first_date_day	14
first_date_day_of_year	318
first_date_month	11
first_date_week	46
season_autumm	1

Showing 11 to 20 of 68 entries

Figure 3.12: The description of a promotion used by forecasting model.

Logic

At the start of the application (with default values) and after each change in the configurator by the user, the simulation process is started.

In the first simulation step, a base example is created. The base example is the example on the basis of which the records entering the input of the predictive model will be simulated. The base example is randomly selected from the historical data of a given product from a given shop. It is a record that describes the sales of the product during a period when there was no promotion, i.e., for example, it is a record describing the sales of product 1234 in shop XYZ during the period from 2017-01-14 to 2017-01-17 in which pricechange = 0. It has the same characteristics as the promotional periods included in the training set.

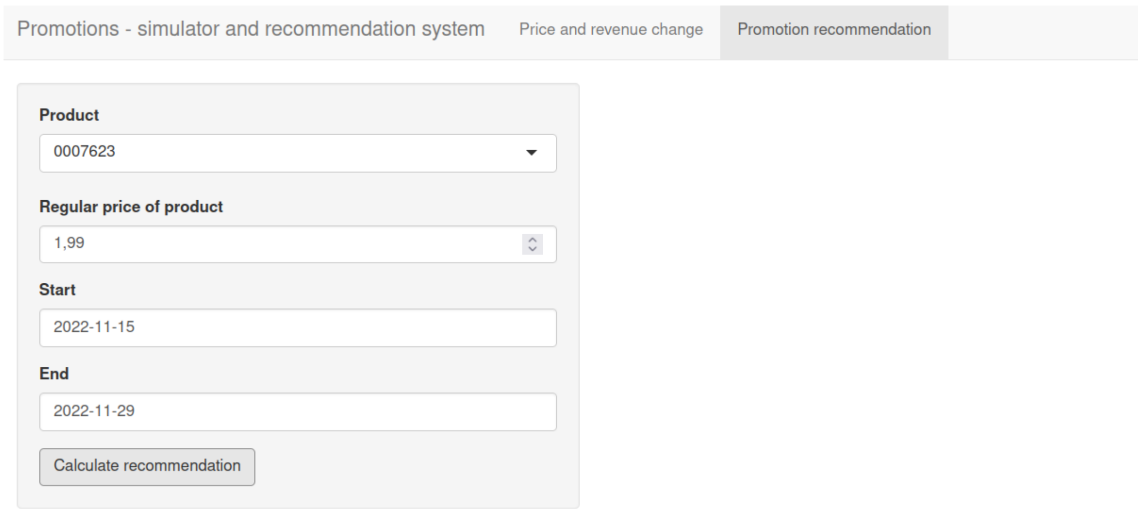
Then the values of all attributes are adjusted to what is indicated in the configurator. The attributes describing the start date of the promotion are modified based on the date selected in the configurator. The attributes relating to the shop and its environment are adapted to the shop selected by the user. Features relating to promotion channels and other promotions are also determined based on the configurator. It should be noted that e.g. the value of the "tv" feature is taken directly from the configurator, while other features, e.g. "or_tv_pd", have to be transformed and determined from several features.

To summarize this step: a base example is created with the feature values indicated by the user.

Based on the base example, 11 examples (data records) are created. Each of these examples corresponds to one price change value (these are values from -0.5 to 0 with a step of 0.05). The examples differ from each other only in the values of the variables 'pricechange' and 'price'. These records are the inputs of the forecasting model. This provides forecasts of the average daily sales of a product for a given promotion configuration and for different price change values. The average daily profit for different values of price change is also calculated on this basis.

The resulting forecasts are used to determine price sensitivity graphs, which are presented to the user in the simulator.

The simulator fulfils its purpose and could be used to study price sensitivity. It allows studying the impact of price changes and different promotion settings on forecast sales.



Product: 0007623
Regular price of product: 1,99
Start: 2022-11-15
End: 2022-11-29

Figure 3.13: Configuration panel of promotion recommendation system.

3.4.2 Promotion recommendation system

The aim of the second iteration was to propose a solution for recommending a promotion. The solution created was to suggest to the user which promotion would be most beneficial to plan.

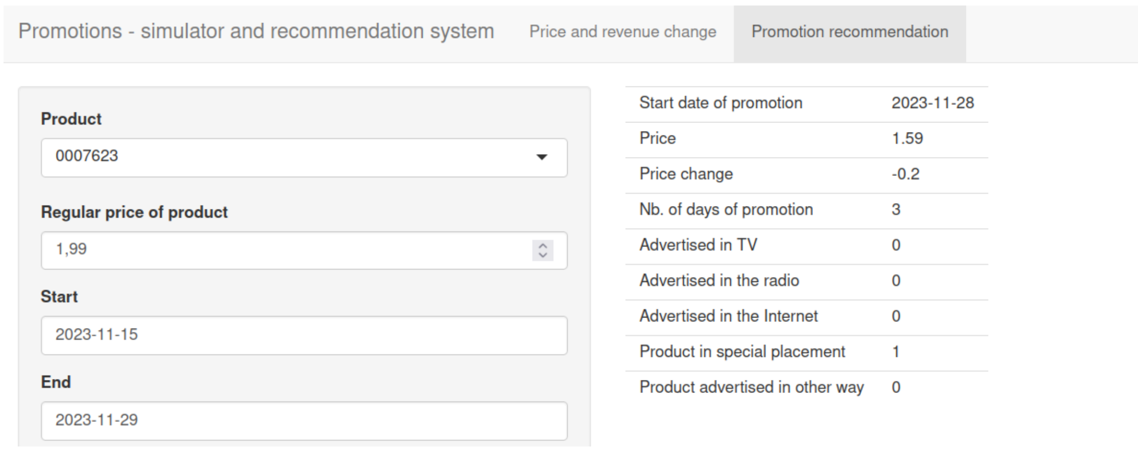
It was intended that the recommendations would relate to nationwide promotions, i.e. promotions for a specific shop were not to be suggested. On entry, the user was to indicate which product the promotion would be for, the regular price of the product and over what period the promotion would run. It was assumed that promotions were planned in the near horizon, so the regular price would be known.

Interface

A prototype application has been proposed that gives the user the possibility to generate promotion recommendations.

The left-hand side of the interface contains configuration fields for the user. The user indicates the product for which they want to create promotions, indicates the regular price of the product, and specifies the period in which the promotion would start. The default period is the next 14 days. The configuration panel is presented in Figure 3.13.

Below the configuration fields, there is a button which, when clicked, the system starts the process of finding the best promotion recommendation for the set parameters. For the default settings, this process takes about 1 minute. While waiting for


Promotions - simulator and recommendation system | Price and revenue change | Promotion recommendation

Product: 0007623

Regular price of product: 1,99

Start: 2023-11-15

End: 2023-11-29

Start date of promotion	2023-11-28
Price	1.59
Price change	-0.2
Nb. of days of promotion	3
Advertised in TV	0
Advertised in the radio	0
Advertised in the Internet	0
Product in special placement	1
Product advertised in other way	0

Figure 3.14: Result of promotion recommendation.

the generated recommendation, a loading icon is displayed. Once a recommendation is found, the app displays a recommendation of a promotional offer. The results can be seen in Figure 3.14.

In the example above, it can be seen that the recommendation system indicates that for a given product with a given regular price, a promotion should start on 2022-11-28 (Monday), where the promotional price would be 1.59. The promotion would last 3 days. Other parameters for the promotion are then given, which by the preset predictive model are necessary to set. The recommendation indicates that the promotion would not be advertised by additional media, and the product should have a special setting in shops.

Logic

The recommendation system works with the same data and model as described in previous chapter. The model used to predict average product sales worked for promotions in a specific shop. The premise of this recommendation system is to suggest general promotions and not for a specific shop. In the proposed solution, a given action was taken into account. The evaluation of the promotion is based on the profit per promotion summed across all shops. The profit is calculated as (promotional price)*(average daily sales forecast).

Recommendation search steps:

1. The system takes configuration data from the user, i.e. product, regular price and promotion search period.

Attributes describing price		Attributes describing date of start of promotion			Attributes describing the shop and its surrounding area			Attributes describing other promotions			Attributes describing promotion channels and combinations of promotion channels		
Price	Price change	Day of the year	...	Weekday	Purchasing power	...	Average salary	Number of promotions	...	Number of promotions with ads	TV	...	TV or radio
1.99	-0.2	56	...	3	99.84	...	4123	120	...	24	0	...	1

Figure 3.15: Recommendation algorithm – example of a base example used in step 2 of algorithm.

2. A base example is searched. It will be the starting point from which the various promotion configurations will be searched. The base example is selected as follows:

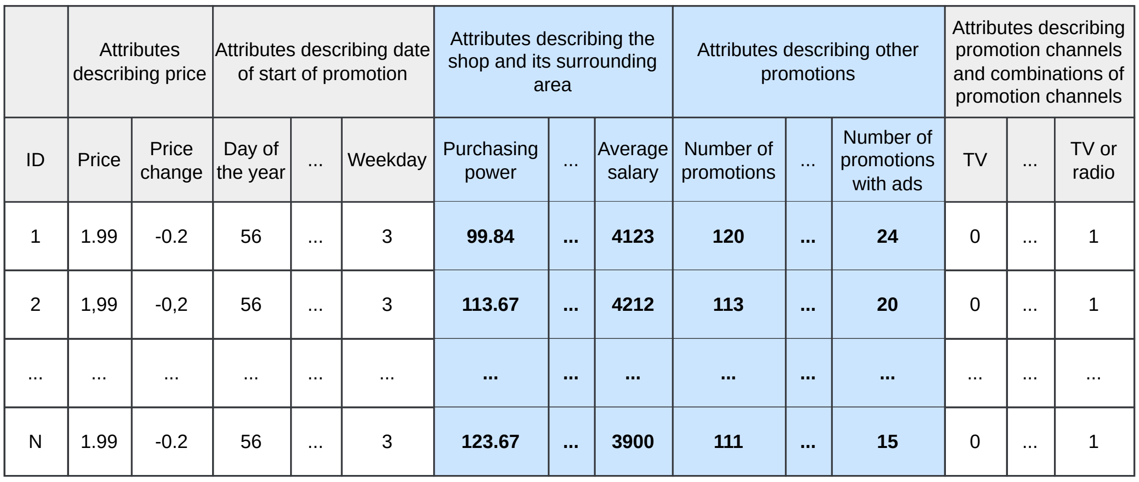
   a) The historical data (training set) is narrowed down to the data describing the selected product, to the records describing promotions (pricechange ¡ 0) and to the characteristics used by the forecasting model.

   b) The centroid of this set is counted – an average value is calculated for all variables.

   c) Using the R function 'FNN::get.knnx', one example is found from the set described in subsection "a)" that is closest to the centroid. The given example becomes our base example.

   An illustrative example of the base example is presented in Figure 3.15.

3. The given base example is then copied as many times as we have shops in the base (suppose we have N shops). We get a new dataset. Each example corresponds to a different shop. The attributes describing the shop and its environment have been changed to describe the specific shop correctly. The attributes describing other promotions that took place during the promotion period (attributes nb_all_promotions, nb_all_promotions_pd, nb_all_promotions_tv_radio_or_internet, nb_all_promotions_tv_radio_internet_or_pd) have been replaced by the average values determined for the shop. The result of this step is a data set in which each example describes promotions with the same parameters but for different shops. This dataset can be seen in Figure 3.16. Features describing other promotions that took place at the time of the promotion have been replaced by average values, as these are store-specific



	Attributes describing price		Attributes describing date of start of promotion			Attributes describing the shop and its surrounding area			Attributes describing other promotions			Attributes describing promotion channels and combinations of promotion channels		
ID	Price	Price change	Day of the year	...	Weekday	Purchasing power	...	Average salary	Number of promotions	...	Number of promotions with ads	TV	...	TV or radio
1	1.99	-0.2	56	...	3	**99.84**	**...**	**4123**	**120**	**...**	**24**	0	...	1
2	1,99	-0,2	56	...	3	**113.67**	**...**	**4212**	**113**	**...**	**20**	0	...	1
...	...	...	...	...	...	**...**	**...**	**...**	**...**	**...**	**...**	...	...	...
N	1.99	-0.2	56	...	3	**123.67**	**...**	**3900**	**111**	**...**	**15**	0	...	1

Figure 3.16: Recommendation algorithm – example of data set describing promotion in each of N shops in step 3 of the algorithm.

features and their values should not be recommended to the user if we are to give recommendations for general promotions.

4. Based on the table created, we calculate the first evaluation value. The projected profit that would be obtained by creating the promotion included in this table is calculated. Each row of the table was run through a predictive model - for each row we obtained some projected average sales value of the product in the shop. The given values were then multiplied by the price of the product and the resulting values were summed. This is the summed projected profit. This resulted in an assessment value, which we will refer to. This step is illustrated by Figure 3.17.

5. A given set was then copied as many times as the number of promotion start dates to be searched (the promotion search period is set by the user). In each set, the features describing the attributes of the promotion start date were modified to match the given date. The result is a list of X datasets and each dataset has N rows - X being the number of days to be searched, N being the number of shops. Suppose we are looking for a promotion to start between 2022-11-17 and 2022-11-19, so we search X = 3 dates. This step is illustrated by Figure 3.18.

6. At this point, we proceed to search through the various promotion settings. The search is performed using climbing. The selected attributes are searched:

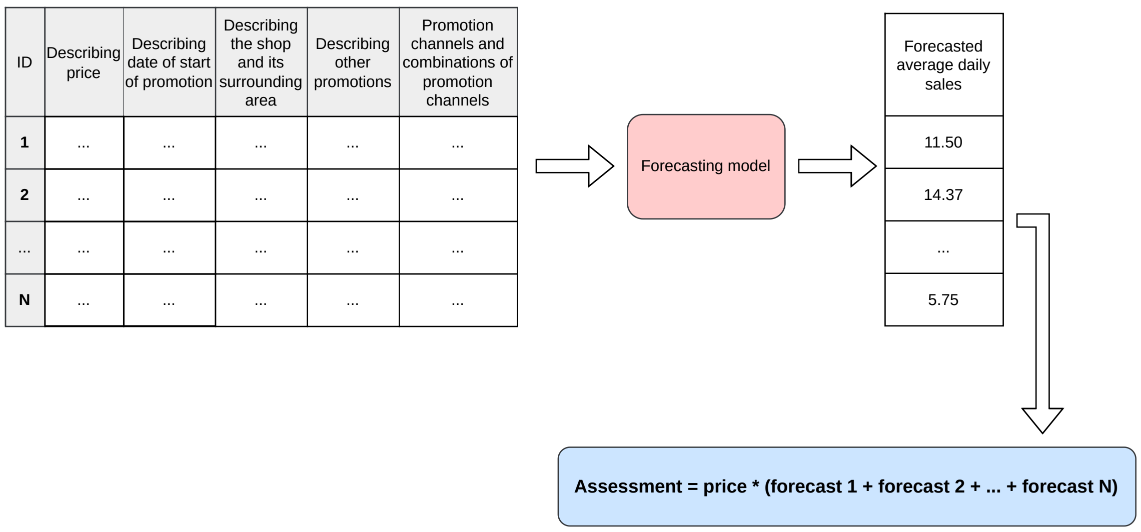

ID
Describing price
Describing date of start of promotion
Describing the shop and its surrounding area
Describing other promotions
Promotion channels and combinations of promotion channels
1
2
...
N
Forecasting model
Forecasted average daily sales
11.50
14.37
...
5.75
Assessment = price * (forecast 1 + forecast 2 + ... + forecast N)

Figure 3.17: Step 4 of recommendation algorithm – calculating assessment value for promotion.

a) "pricechange" – search between -0.5 and 0 with step 0.05
b) "days_count" – search from 1 to 8 with step 1
c) "tv" – searched values are 0 and 1
d) "radio" – searched values are 0 and 1
e) "internet" – searched values are 0 and 1
f) "spec_placement" – searched values are 0 and 1
g) "promo_designer" – searched values are 0 and 1

For each date, further datasets are created in which all attributes retain their existing values and only the value of one of the above attributes is changed. This is presented in Figure 3.19.

Having the modified data sets for each of them, we calculate the assessment value, i.e. the total profit value of the promotion in question. The calculation of the rating is done in the same way as was presented in step 4.

7. After calculating the rating, we select the set for which the highest rating was obtained (Figure 3.20). The given set defines the promotions in each shop, on the specific promotion start date studied and with the one attribute that was searched changed, where the attribute gets a specific value. After this step, we assume that the selected date will be the recommended promotion start

Data set for the date **2022-11-17** describing the promotion at each of the N shops

ID	Describing price	Describing date of start of promotion			Describing the shop and its surrounding area	Describing other promotions	Promotion channels and combinations of promotion channels
	...	Day of the year	...	Weekday	...	...	...
1	...	**321**	**...**	**4**	...	...	...
...	...	**321**	**...**	**4**	...	...	...
N	...	**321**	**...**	**4**	...	...	...

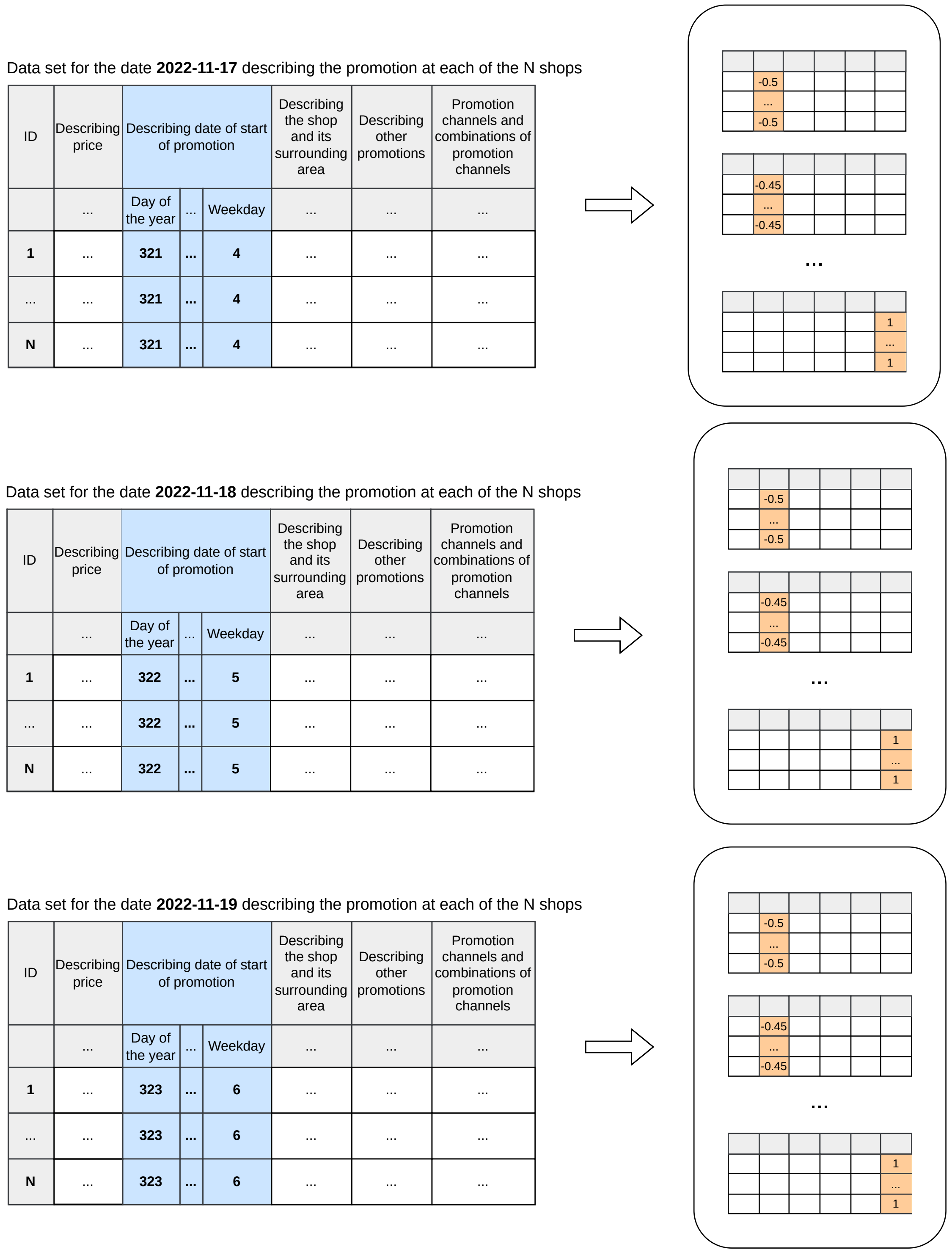
Data set for the date **2022-11-18** describing the promotion at each of the N shops

ID	Describing price	Describing date of start of promotion			Describing the shop and its surrounding area	Describing other promotions	Promotion channels and combinations of promotion channels
	...	Day of the year	...	Weekday	...	...	...
1	...	**322**	**...**	**5**	...	...	...
...	...	**322**	**...**	**5**	...	...	...
N	...	**322**	**...**	**5**	...	...	...

Data set for the date **2022-11-19** describing the promotion at each of the N shops

ID	Describing price	Describing date of start of promotion			Describing the shop and its surrounding area	Describing other promotions	Promotion channels and combinations of promotion channels
	...	Day of the year	...	Weekday	...	...	...
1	...	**323**	**...**	**6**	...	...	...
...	...	**323**	**...**	**6**	...	...	...
N	...	**323**	**...**	**6**	...	...	...

Figure 3.18: Step 5 of recommendation algorithm – duplicating data set for each possible date of start of promotion.


Data set for the date **2022-11-17** describing the promotion at each of the N shops

ID	Describing price	Describing date of start of promotion			Describing the shop and its surrounding area	Describing other promotions	Promotion channels and combinations of promotion channels
	...	Day of the year	...	Weekday	...	...	...
1	...	**321**	**...**	**4**	...	...	...
...	...	**321**	**...**	**4**	...	...	...
N	...	**321**	**...**	**4**	...	...	...

Data set for the date **2022-11-18** describing the promotion at each of the N shops

ID	Describing price	Describing date of start of promotion			Describing the shop and its surrounding area	Describing other promotions	Promotion channels and combinations of promotion channels
	...	Day of the year	...	Weekday	...	...	...
1	...	**322**	**...**	**5**	...	...	...
...	...	**322**	**...**	**5**	...	...	...
N	...	**322**	**...**	**5**	...	...	...

Data set for the date **2022-11-19** describing the promotion at each of the N shops

ID	Describing price	Describing date of start of promotion			Describing the shop and its surrounding area	Describing other promotions	Promotion channels and combinations of promotion channels
	...	Day of the year	...	Weekday	...	...	...
1	...	**323**	**...**	**6**	...	...	...
...	...	**323**	**...**	**6**	...	...	...
N	...	**323**	**...**	**6**	...	...	...

Figure 3.19: Step 6 of recommendation algorithm – creating datasets for climbing process of searching the best configuration of promotion.

date and the other dates will no longer be searched. We also recognize that an attribute that has been changed in a given set will take a given value and this value will not be changed in further steps either.

8. The maximum score obtained in the previous step is recognised as the new baseline score. After the previous step, as a result of climbing, we have found the corresponding value of one attribute. The given attribute is removed from the list of searched attributes and the climbing process continues. Again, the set selected in the previous step as the one with the maximum rating is modified in such a way that the value of one attribute in each set is changed. This is illustrated by Figure 3.21. We are already considering the sets with the date set in the previous step.

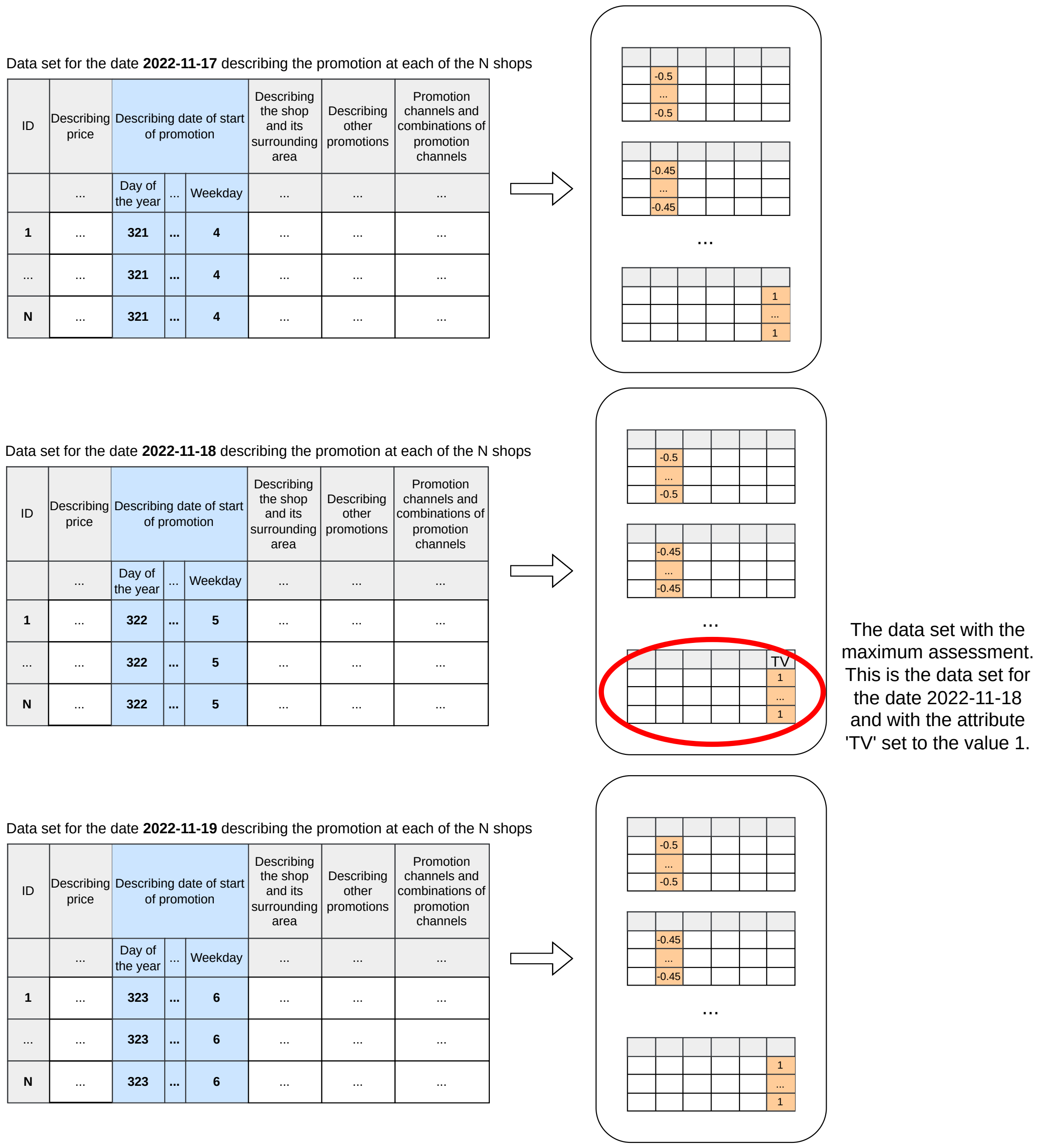
9. The set with the maximum score is selected. The given set indicates which attribute and its value gives the maximum rating. If this rating is greater than the current baseline rating, it means that a better promotion has been achieved than in the previous step. In this case, the climbing process continues – a return to step number 8 is made. If a rating less than or equal to the baseline rating is obtained, we proceed to step 10.

10. The climbing process has been completed. A combination of attribute values has been obtained that determines the recommended promotion. The given attributes are returned to the user. This step is presented in Figure 3.22.

3.4.3 Extended promotion recommendation system

The aim of the third iteration was to expand the module presented in first promotion recommendation system. The extension was to consist of adding additional constraints that would control whether the promotion recommendation returned by the system is sufficiently good in terms of other promotion indicators (not just in terms of the number of units sold).

From the user's point of view, they should have the ability to receive a promotion recommendation not only to maximize the average daily sales of the product during the promotion period, but also to meet additional constraints. The constraints would be ensuring that the average basket value of the promoted product does not fall below a specified level and/or the average number of customers falls within a specified range. This can prevent a situation where a promotion is recommended that maximizes the


Data set for the date **2022-11-17** describing the promotion at each of the N shops

ID	Describing price	Describing date of start of promotion			Describing the shop and its surrounding area	Describing other promotions	Promotion channels and combinations of promotion channels
	...	Day of the year	...	Weekday	...	...	...
1	...	**321**	**...**	**4**	...	...	...
...	...	**321**	**...**	**4**	...	...	...
N	...	**321**	**...**	**4**	...	...	...

Data set for the date **2022-11-18** describing the promotion at each of the N shops

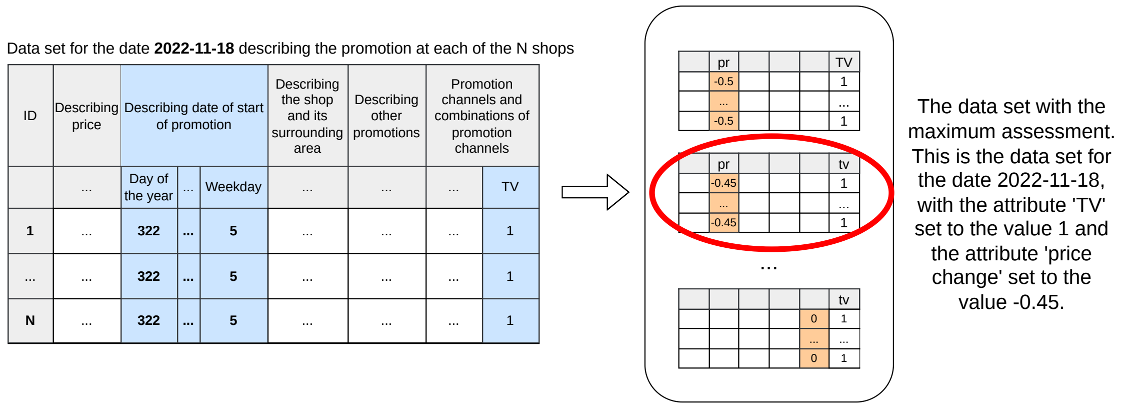
ID	Describing price	Describing date of start of promotion			Describing the shop and its surrounding area	Describing other promotions	Promotion channels and combinations of promotion channels
	...	Day of the year	...	Weekday	...	...	...
1	...	**322**	**...**	**5**	...	...	...
...	...	**322**	**...**	**5**	...	...	...
N	...	**322**	**...**	**5**	...	...	...

Data set for the date **2022-11-19** describing the promotion at each of the N shops

ID	Describing price	Describing date of start of promotion			Describing the shop and its surrounding area	Describing other promotions	Promotion channels and combinations of promotion channels
	...	Day of the year	...	Weekday	...	...	...
1	...	**323**	**...**	**6**	...	...	...
...	...	**323**	**...**	**6**	...	...	...
N	...	**323**	**...**	**6**	...	...	...

-0.5
...
-0.5
-0.45
...
-0.45
...
1
...
1
TV
The data set with the maximum assessment. This is the data set for the date 2022-11-18 and with the attribute 'TV' set to the value 1.

Figure 3.20: Step 7 of recommendation algorithm – selecting the dataset with highest assessment in first iteration of search procedure.

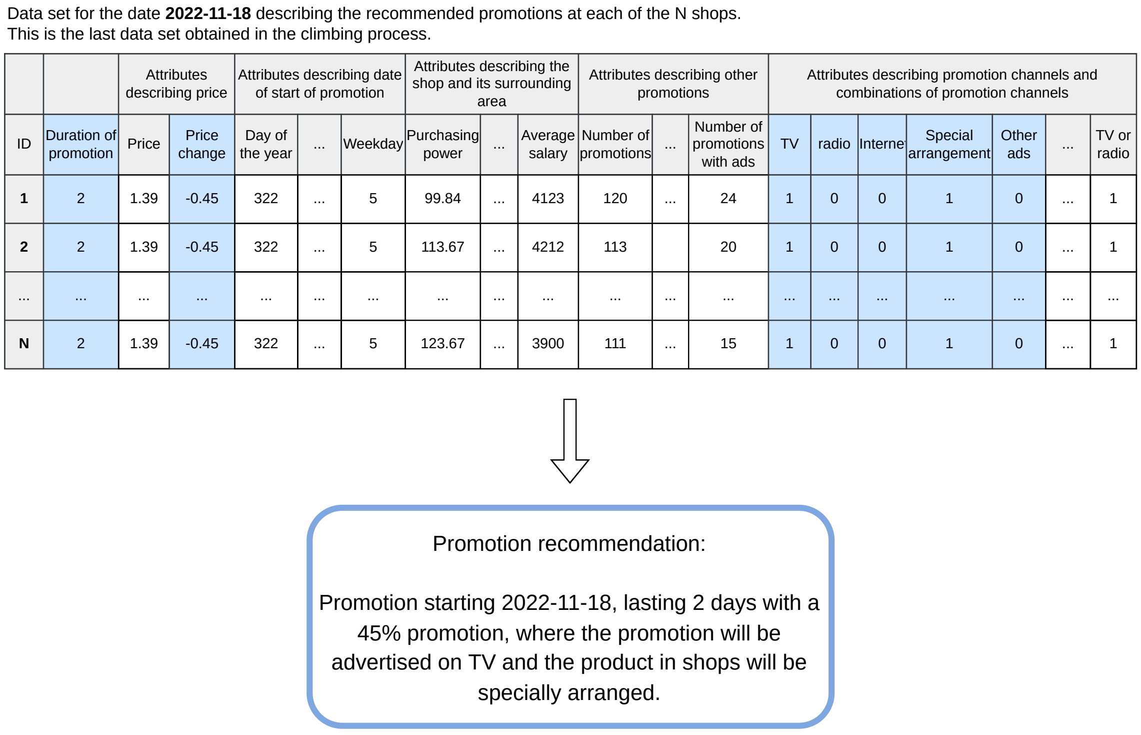


Data set for the date **2022-11-18** describing the promotion at each of the N shops

ID	Describing price	Describing date of start of promotion			Describing the shop and its surrounding area	Describing other promotions	Promotion channels and combinations of promotion channels	
	...	Day of the year	...	Weekday	...	...	...	TV
1	...	**322**	**...**	**5**	...	...	...	1
...	...	**322**	**...**	**5**	...	...	...	1
N	...	**322**	**...**	**5**	...	...	...	1

The data set with the maximum assessment. This is the data set for the date 2022-11-18, with the attribute 'TV' set to the value 1 and the attribute 'price change' set to the value -0.45.

Figure 3.21: Step 8 of recommendation algorithm – next iteration of search procedure.



Data set for the date **2022-11-18** describing the recommended promotions at each of the N shops.
This is the last data set obtained in the climbing process.

		Attributes describing price		Attributes describing date of start of promotion			Attributes describing the shop and its surrounding area			Attributes describing other promotions			Attributes describing promotion channels and combinations of promotion channels						
ID	Duration of promotion	Price	Price change	Day of the year	...	Weekday	Purchasing power	...	Average salary	Number of promotions	...	Number of promotions with ads	TV	radio	Interne	Special arrangement	Other ads	...	TV or radio
1	2	1.39	-0.45	322	...	5	99.84	...	4123	120	...	24	1	0	0	1	0	...	1
2	2	1.39	-0.45	322	...	5	113.67	...	4212	113		20	1	0	0	1	0	...	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
N	2	1.39	-0.45	322	...	5	123.67	...	3900	111	...	15	1	0	0	1	0	...	1

Promotion recommendation:

Promotion starting 2022-11-18, lasting 2 days with a 45% promotion, where the promotion will be advertised on TV and the product in shops will be specially arranged.

Figure 3.22: Step 10 of recommendation algorithm – result of recommendation system.

number of units sold, but in terms of other indicators, it is a weak promotion (e.g. the average basket value of the promoted product is very low because customers will only buy that product without making larger purchases).

The study used two additional indicators – the forecasted average daily number of customers in the store during the promotion period and the forecasted average basket value for the product. For these two indicators, predictive models were created as part of a previous research project, which based on the given attributes describing the promotion, predicted the value of the indicator. The models used the same conditional attributes as the model predicting the average number of units sold during the promotion period. This model was used in the research for iteration 1 and iteration 2.

Interface

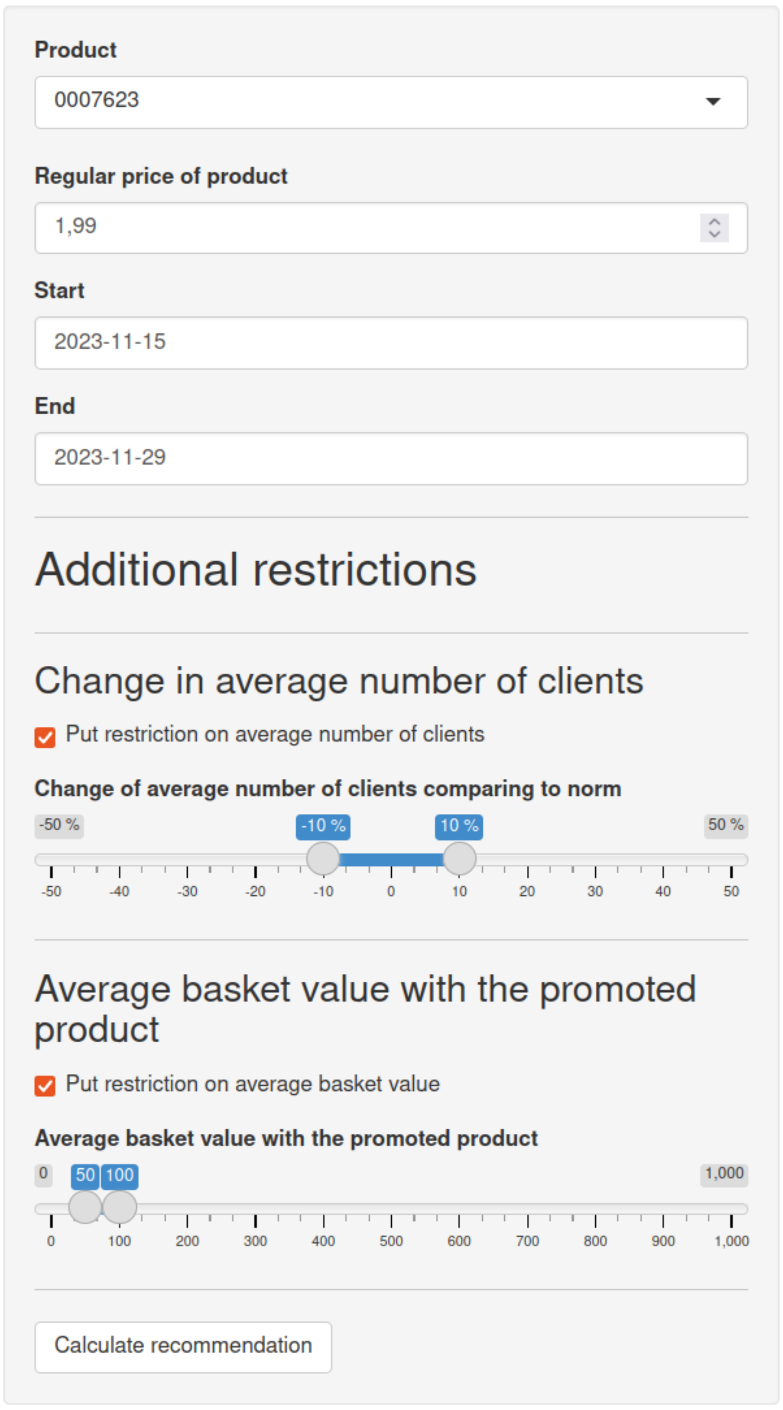
In the promotion recommendation module, a new section - "Additional restrictions" - has been added to the user configuration panel. It is presented in Figure 3.23.

By using the introduced controls, the user can indicate whether they want to include additional constraints on the forecasted values of other promotion indicators in the recommendation process. If they want to add such constraints, they check the checkbox and use the sliders to select the values that interest them.

The first slider concerns setting a constraint on the average number of customers in the store during the promotion period. An example of setting the value relative to the norm is shown here, where the norm is the average number of customers calculated based on historical data. The user can indicate how the number of customers should change during the promotion period. For example, if the user sets the range from -10% to 10%, this means that the forecasted value of the average number of customers for the proposed promotion should not deviate from the norm and the forecast should be between 0.9norm and 1.1norm.

The second slider concerns setting a constraint on the average value of the basket with the promoted product. An example of setting a value directly is shown here. The user can indicate that for the proposed promotion, the forecasted average value of the basket with the specified product should be, for example, in the range of 100 to 300 PLN.

The other parts of the configuration panel work as described in iteration 2. After selecting all settings, with or without optional additional constraint settings, the user clicks the "Calculate recommendation" button, and the system runs the recom-



Product

0007623

Regular price of product

1,99

Start

2023-11-15

End

2023-11-29

Additional restrictions

Change in average number of clients

✓ Put restriction on average number of clients

Change of average number of clients comparing to norm

-50 % | -10 % | 10 % | 50 %

Average basket value with the promoted product

✓ Put restriction on average basket value

Average basket value with the promoted product

0 | 50 | 100 | 1,000

Figure 3.23: "Additional restrictions" in configuration panel of promotion recommendation system.

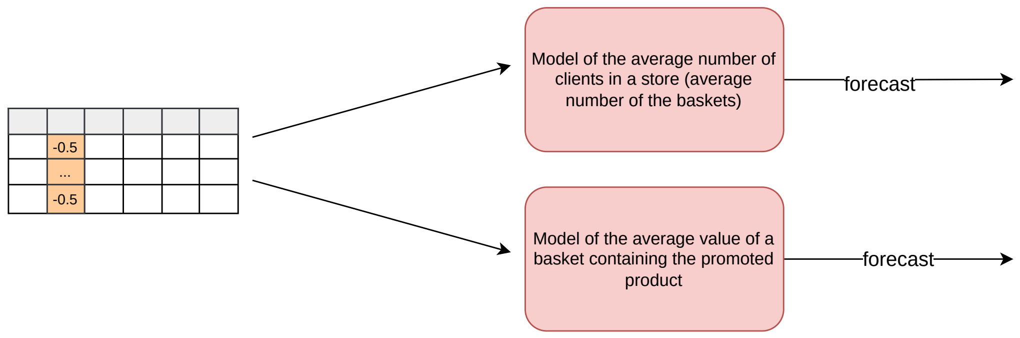

-0.5
...
-0.5
Model of the average number of clients in a store (average number of the baskets)
forecast
Model of the average value of a basket containing the promoted product
forecast

Figure 3.24: Promotion recommendation system with additional restrictions for other indications – obtaining additional forecasts.

mendation algorithm and returns information to the user regarding the proposed promotion parameters.

Logic

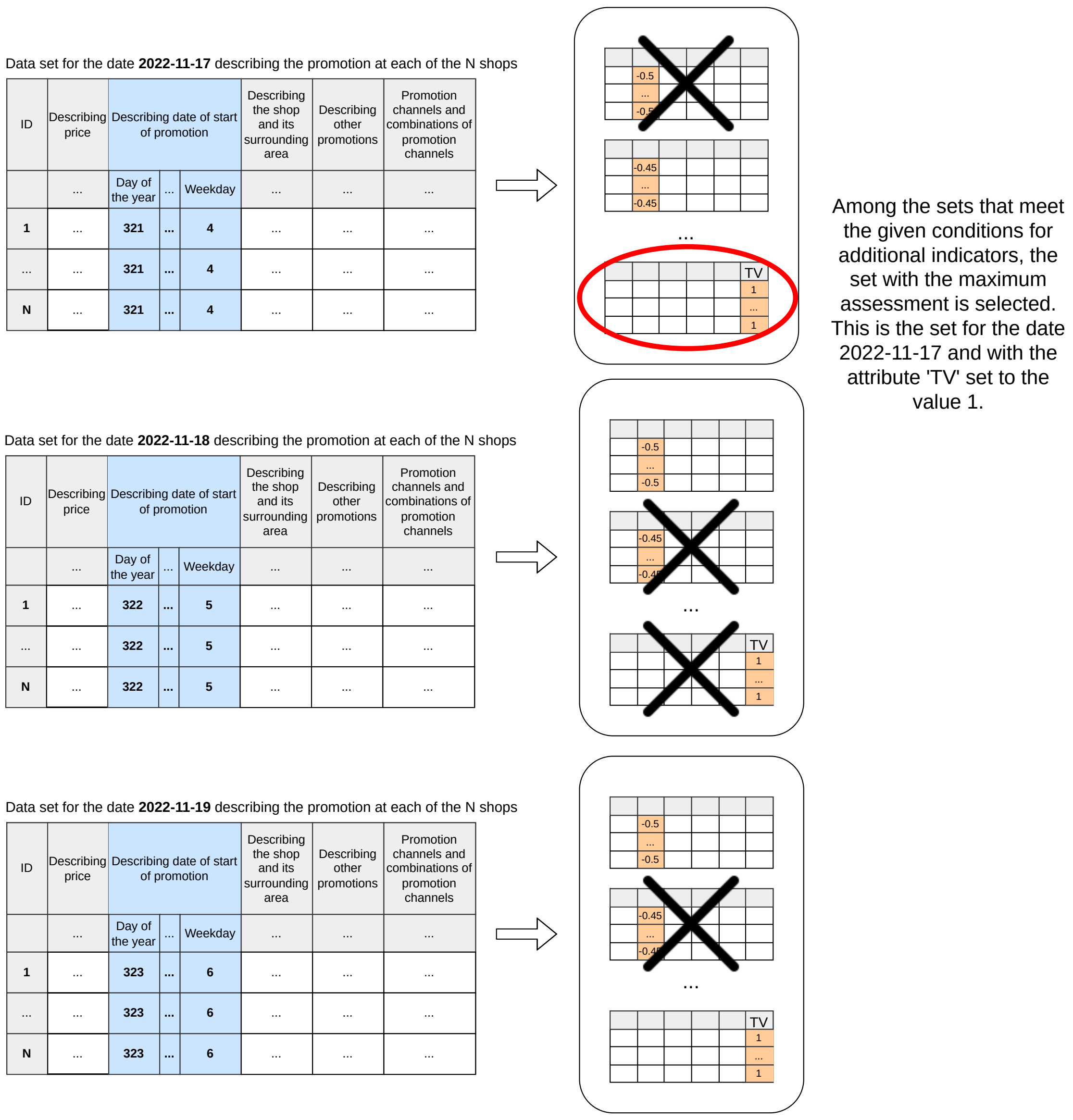
The logic of the algorithm in this iteration has been slightly extended. Therefore, the entire algorithm will not be presented here, only the introduced changes will be shown.

The change occurs in step 7 of the presented algorithm, after calculating the score for each considered promotion start date and for each value of the changed conditional attribute. Before selecting the set with the maximum score, additional constraints imposed by additional indicators are checked. For each set considered in step 6 of the algorithm, forecasts for 2 indicators are calculated - the average number of customers in the store and the average value of the basket with the promoted product. This is illustrated in Figure 3.24.

The obtained forecasts are then compared with the ranges specified by the user. If the forecast for a particular set does not meet the specified constraints, such a set is removed and does not participate in the selection of the set with the maximum score. Schematically, it looks like in Figure 3.25.

The scheme from Figure 3.25 differs from the scheme presented in Figure 3.20 only in that some sets are removed from consideration if the set (i.e. combination of promotion parameters) does not meet at least one constraint set by the user.

The same difference is repeated in the subsequent steps of the algorithm, i.e., in step 8 what is presented in Figure 3.26. While considering the next combinations of conditional attributes, forecasts for additional indicators are made each time. If a


Data set for the date **2022-11-17** describing the promotion at each of the N shops

ID	Describing price	Describing date of start of promotion			Describing the shop and its surrounding area	Describing other promotions	Promotion channels and combinations of promotion channels
	...	Day of the year	...	Weekday	...	...	...
1	...	**321**	**...**	**4**	...	...	...
...	...	**321**	**...**	**4**	...	...	...
N	...	**321**	**...**	**4**	...	...	...

Data set for the date **2022-11-18** describing the promotion at each of the N shops

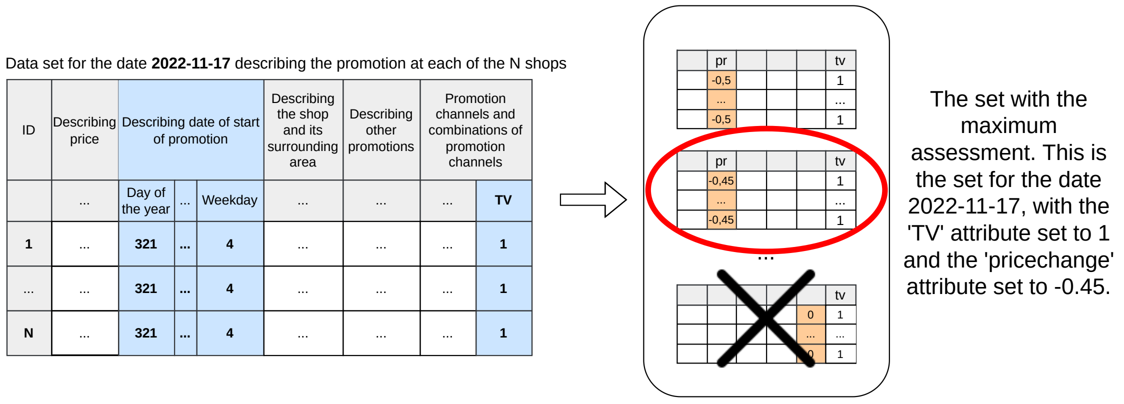
ID	Describing price	Describing date of start of promotion			Describing the shop and its surrounding area	Describing other promotions	Promotion channels and combinations of promotion channels
	...	Day of the year	...	Weekday	...	...	...
1	...	**322**	**...**	**5**	...	...	...
...	...	**322**	**...**	**5**	...	...	...
N	...	**322**	**...**	**5**	...	...	...

Data set for the date **2022-11-19** describing the promotion at each of the N shops

ID	Describing price	Describing date of start of promotion			Describing the shop and its surrounding area	Describing other promotions	Promotion channels and combinations of promotion channels
	...	Day of the year	...	Weekday	...	...	...
1	...	**323**	**...**	**6**	...	...	...
...	...	**323**	**...**	**6**	...	...	...
N	...	**323**	**...**	**6**	...	...	...

Among the sets that meet the given conditions for additional indicators, the set with the maximum assessment is selected. This is the set for the date 2022-11-17 and with the attribute 'TV' set to the value 1.

Figure 3.25: Step 7 of promotion recommendation system with additional restrictions for other indications – selecting sets that meet the specified constraints.


Data set for the date **2022-11-17** describing the promotion at each of the N shops

ID	Describing price	Describing date of start of promotion			Describing the shop and its surrounding area	Describing other promotions	Promotion channels and combinations of promotion channels	
	...	Day of the year	...	Weekday	...	...	...	**TV**
1	...	**321**	**...**	**4**	...	...	...	**1**
...	...	**321**	**...**	**4**	...	...	...	**1**
N	...	**321**	**...**	**4**	...	...	...	**1**

pr
tv
-0,5
...
1
-0,45
0
The set with the maximum assessment. This is the set for the date 2022-11-17, with the 'TV' attribute set to 1 and the 'pricechange' attribute set to -0.45.

Figure 3.26: Step 8 of promotion recommendation system with additional restrictions for other indications – selecting sets that meet the specified constraints.

given combination (i.e., a given promotion) does not meet the set constraints, it is removed from consideration.

Tests of recommendations

As part of the research, the generated promotion recommendations were tested. For each product in the studied product group, the best promotion was searched for in two periods: from 2022-10-31 to 2022-11-30 and from 2023-01-10 to 2023-04-10 (the recommended promotion was to start in the given periods). Recommendations depending on the regular price of the product were also examined.

The tests show that the recommendation algorithm suggests different promotions depending on the input values. This means that the algorithm can be applied in real-world conditions, as it will not always point the user to the same promotion regardless of the given input conditions.

The running time of the algorithm varies from 52 seconds to about 4 minutes. The running time depends on the period searched.

Different promotion start days and different days of the week were recommended. However, in no case did the algorithm give a recommendation to start the promotion on a Monday. Different promotion durations are also recommended: 2, 3, 4, 5 or 8 days. Different promotion channels and special product positioning are also recommended.

3.5 Recommendations using survival action rules

The previous section showed how to use predictive XGBoost models to recommend the best promotion and to simulate price changes. This section will elaborate on the topic.

The proposed promotion quality indicators show how good a promotion is. The indicator is determined as the average of the values for each day of the promotion. Here we will look at the given issue from a different perspective. We want to conduct an analysis of where the promotion indicator falls below a specified threshold during the time of promotion. We will then want to find changes that could be applied to keep the promotion indicator above the threshold for a longer number of days, and preferably for the entire duration of the promotion. Such suggestions for changes can also be considered as a promotion simulator, but taking into account the granularity up to days rather than for the entire promotion period.

Survival action rules will be used for this purpose. This is an original method co-developed by the author. To the best of our knowledge, this is a first method for generating action rules for censored data and survival analysis. In order to understand the issue, the theory associated with the algorithm will be presented here.

3.5.1 Theory for Survival Action Rules

Survival Analysis

Survival analysis involves data mining methods in which the dependent variable describes the elapsed time until a certain event occurs [144]. The dependent variable could be, for example, the time elapsed from the performance of a medical procedure to the recurrence of the disease. Very often the aim of survival analysis is predicting the time elapsed until a certain event occurs.

In the dataset, for which survival analysis should be performed, there need to be two attributes – *survival time* and *survival status*. The dataset containing survival information can be written in a more formal form as $D(A, T, \delta)$, where A is a set of conditional attributes, T is a survival time and δ describes survival status. Survival status is a binary attribute. It is assumed that if the event has occurred, the survival status is 1. Otherwise, it is equal to 0. If the i-th observation belongs to the examples for which the studied event did not occur ($\delta_i = 0$) then T_i determines the observation time. Otherwise ($\delta_i = 1$), T_i is the elapsed time to the event [170].

In survival datasets there are *censored* observations. This means that we have incomplete information about the time of occurrence of the event. For observations for which the survival status is 0, we only know how much the survival time was at least, but we do not know when or if the event occurred at all. This research focused on data containing right-censored observations. It means that for censored observations the actual time of occurrence of the event is longer than the observed time.

Censoring makes it impossible to use classical regression methods. For this reason, dedicated methods are used for survival analysis. From the statistical methods the Kaplan-Meier estimator and the log-rank test can be highlighted.

The Kaplan-Meier estimator (KM) [102] is an estimator of the survival function $\hat{S}(t)$, which determines the probability of surviving longer than a given time t. This function is a non-increasing function ($\hat{S}(t_1) \geqslant \hat{S}(t_2) \iff t_1 \leqslant t_2$), which for time $t = 0$ takes the value 1 ($\hat{S}(0) = 1$), and at t going to infinity is equal to 0 ($\lim_{t \to +infty} \hat{S}(t) = 0$).

The KM estimator assumes that censoring is independent of survival time. A discrete survival time [144] is also assumed. The Kaplan-Meier estimator for time t is described by the formula 3.1.

$$\hat{S}(t) = \prod_{j:t_j \leqslant t} \frac{(r_j - d_j)}{r_j}, \text{ dla } 0 \leqslant t \leqslant t^+ \tag{3.1}$$

In the definition of 3.1, r_j is the number of observations at risk and d_j is the number of occurrences of the event under study at time t_j, where t_j is the successive times of occurrence of the event such that $t_j < t_{j+1}$. t^+ is the maximum time of occurrence of the event. At-risk observations are those examples for which the observation time is greater than t_j.

The results of the Kaplan-Meier estimator are often presented graphically by so-called *survival curves* (or *KM curves*). An example of such a graph is shown in Figure 3.27. On the X-axis is the time t, and on the Y-axis is the survival probability value. Survival probability means the probability of non-occurrence of the event defined by survival status. KM curves are often used to compare the results of the Kaplan-Meier estimator for different groups of observations. For example, if one KM curve is below another, it means that it has a lower probability of survival.

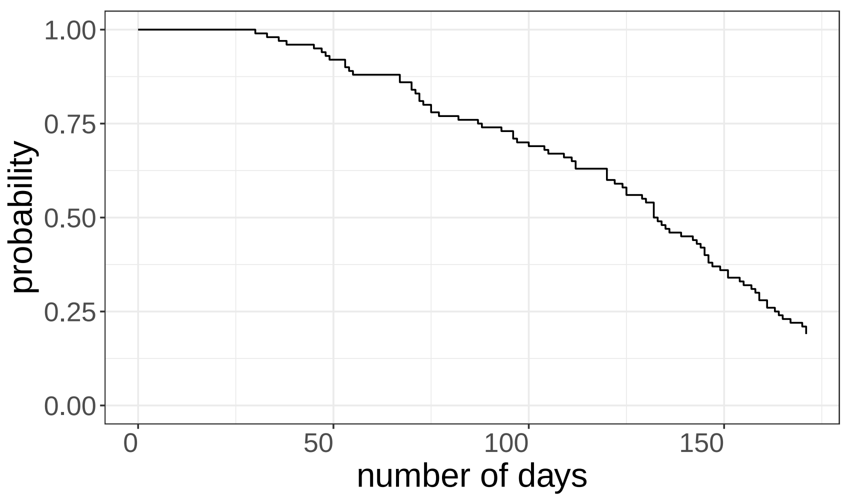


Figure 3.27: Survival curve for exemplary dataset.

The log-rank test is a χ^2 test. It is used to verify that KM curves are significantly different from each other. It is a non-parametric test. The null hypothesis is that the survival curves are identical.

Rule mining

Rule induction is classified as a machine learning method [140]. Rule-based methods can be used for data mining, representation of the results of a learning algorithm, and prediction[60].

Each rule is of the form:

$$\text{IF } \varphi \text{ THEN } \psi$$

A rule is composed of a premise (φ) and a conclusion (ψ). The premise tells us what conditions must be met for the conclusion to occur. The premise consists of *elementary conditions*, which can be represented as A_i *op* V, where A is a conditional attribute, *op* is a relational operation and V is the scope of the condition.

A survival rule is a formula that tells what conditions an example must satisfy in order for the estimation of its survival function to equal the estimator found in the conclusion. The conclusion of the rule is the KM estimator of the $\hat{S}$ survival function. Survival rule r can be presented as follows:

$$\text{IF } w_1 \wedge w_2 \wedge \ldots \wedge w_n \text{ THEN } \hat{S}$$

The conditions found in the premise of the rule have been labeled as $w_1, w_2, \ldots, w_n$, where n is the number of all conditions found in the premise.

Algorithm for survival rule induction is presented in the paper [170].

Action mining

Action mining [49] is a domain that focus on methods for finding changes that should be made in order to change class assigment of an example from the dataset. The recommendations can present what should be done in order to e.g. change the class of a patient from “sick patient” to “healthy patient”.

Action rules are a part of action mining domain. They are typically defined for the classification problem. Action rules are formulas that describe transitions between variable values that cause a change in the conclusion. For a classification problem, an action rule shows how attribute values should change so that an example is assigned to a different decision class. An action rule has a form:

$$\text{IF } w_{1S} \to w_{1T} \wedge w_{2S} \to w_{2T} \wedge \ldots \wedge w_{nS} \to w_{nT} \text{ THEN } C_S \to C_T \ .$$

The premise of an action rule consists of the conjunction of elementary actions. The elementary action $w_{iS} \to w_{iT}$ represents a change in the value of the attribute a_i and consists of the premise of the elementary action w_S specifying the source range of values of this attribute and the conclusion of the elementary action w_T specifying the target range of these values.

The following types of elementary actions are distinguished:

- elementary action of the form: $w_{iS} \to w_{iT}$, where $w_{iS} \neq w_{iT}$, which represents the change in the value of the attribute a_i,
- elementary arbitrary action: $w_{iS} \to ANY$, where the target range of the attribute values is not specified,
- elementary sustaining action: $w_{iS} \to w_{iT}$, where $w_{iS} = w_{iT}$, which represents the preservation of the source range of attribute values.

The action rule containing source (left) and target (right) parts can be seen as a composition of two different rules (called source and target).

Survival action rules

Survival action rules are a combination of typical action rules generated for classification problems and survival rules. This kind of rules indicate the action that should be made (described in premise) in order to change survival curve, which is included in the conclusion of the rule.

The survival action rule can be presented as:

$$\text{IF } w_{1S} \rightarrow w_{1T} \wedge w_{2S} \rightarrow w_{2T} \wedge \ldots \wedge w_{nS} \rightarrow w_{nT} \text{ THEN } \hat{S}_S \rightarrow \hat{S}_T \ .$$

The illustrative example of survival action rule could have the following form:

IF $(drug, (-inf, 10) \rightarrow [12, 20)) \wedge (recovery, (-inf, 7) \rightarrow [30, inf)$ THEN $\hat{S}_S \rightarrow \hat{S}_T$

where the estimators $\hat{S_S}$ and $\hat{S_T}$ describe the probabilities of patient survival. This survival action rules present how the dose of drug and recovery time should change in order to change KM curve.

3.5.2 Induction of survival action rules

The innovative new survival action rule induction algorithm will be presented below. This is a covering rule induction algorithm [61]. The main function of the algorithm is presented in Algorithm 1. In each pass of the loop, one rule is generated. The creation of such a rule consists of two stages: the growth (specialization) of the rule (line 5) and pruning (generalization, line 6). At the end of the loop run, the examples that the new rule covers are removed from the set of examples not yet covered. If the creation of a rule results in an empty rule, i.e. one that does not have any conditions in the premise, it will not be added to the final set of all rules (check in line 7). This means that it was no longer possible to find a new action rule. Such a rule also causes the entire uncovered set of examples to be covered, thus the induction process ends.

The proposed algorithm can generate target survival curves in different positions. The user can indicate whether they are interested in the target survival curves being improving (experiment type *BETTER*), worsening (experiment type *WORSE*) or arbitrary relative to the source curve (experiment type *ANY*). The log-rank measure [22], which is used to indicate maximally distant curves, is symmetric, meaning that it does not take into account th relative position of the KM curves. The area under the survival curves and the survival probability for the middle observation time for the KM estimators are therefore examined when developing a new rule. If an improving curve is sought, then 2 conditions must be met:

- The area under the graph of the target curve must be greater than the area under the graph of the reference curve.
- The probability of survival for the middle time is higher than the probability obtained for the KM estimator of examples covered by the source (reference) rule.

When looking for a survival rule, the opposite conditions will be checked.

Algorithm 1 An action rule induction algorithm for censored data

```
Input:
  $D(A, T, \delta)$ – a data set described by attributes $A$, observation time $T$ and survival status $\delta$
  $\mu$ – the minimum number of examples not yet covered that the new rule must cover
  $\xi$ – the maximum percentage of examples common to both rules
  $\rho$ – the maximum rule coverage
  $\tau$ – flag specifying whether the target rule is an improving rule ($\tau$ = 'BETTER'), worsening ($\tau$ = 'WORSE') or any ($\tau$ = 'ANY')
  $A_{stable}$ – a set of stable attributes from the set $A$
Output: $R$ – a set of experiential rules for action
 1: function InductionOfSurvivalActionRules($D$, $\mu$, $\tau$, $A_{stable}$)
 2:     $R \leftarrow \varnothing$
 3:     $D_u \leftarrow D$                              ▷ set of examples that are not covered
 4:     while $D_u \neq \varnothing$ do
 5:         $r \leftarrow$ Specialise($D, D_u, \mu, \tau, \rho, A_{stable}$)
 6:         $r \leftarrow$ Prune($D, r, \tau, \xi, \rho$)
 7:         if $r \neq \varnothing$ then $R \leftarrow R \cup \{r\}$
 8:         $D_c \leftarrow$ CoveredExamples($r, D$)  ▷ function returns examples from the set $D$ covered by the rule $r$
 9:         $D_u \leftarrow D_u \backslash D_c$
10:     end while
11:     return $R$
12: end function
```

The specialisation of the rule is presented in Algorithm 2. Growth involves greedily adding more actions to the initially empty premise of the rule being created. In each step, the best elementary condition that can be added to the left side of the rule (line 5) and the best counter-condition, that is, the best elementary condition that can be added to the right side of the action rule (line 9) are found. The action, created from these two conditions, is then added to the rule premise (lines 10-13). If the best elementary condition for the left side is not found then the rule growth stage is completed. The line 6 shows the addition of the found best elementary condition to the set of already reviewed conditions. This is because if a counter-condition is not found for such a condition, the action with this condition will not be added to the premise of the rule. On the next run of the loop, this condition cannot be reviewed again in search of the best condition for the left rule, because otherwise the same condition would be indicated again. For this reason, conditions already reviewed are saved in the W_{ignore} set, in order to be ignored when reviewing new conditions.

Finding the best elementary condition (line 5) involves looking for an elementary condition that, when added to the premise of the left-hand side of an action rule, will produce a survival rule of the best quality. In other words, the condition that maximizes the value of the log-rank statistic between the subset of examples covered and not covered by such a rule is sought. If several conditions result in the best value of the log-rank measure, the one that most increases the coverage of the set of examples not yet covered is selected.

Finding the best counter-condition is presnted in line line 9. The counter-condition is an elementary condition w_{iT} for attribute a_i, which maximizes the value of the log-rank test between the subset covered by the so far determined premise of the source part of the action rule with added condition w_{iS} and the subset of examples covered by the so far determined premise of the target part of the action rule with added condition w_{iT}.

After the specialisation of the rule, the pruning step (generalisation of the created rule) is executed. This procedure iteratively removes actions from the premise and checks whether the value of the log-rank measure obtained for the new rule between examples covered by the left side of the rule and the right side improves or stays the same. In the same loop run, the effect of changing a given elementary action to an arbitrary action ($w_{iS} \rightarrow ANY$) is also examined.

After each run of the inner loop, the action whose absence causes the greatest increase in quality of the action rule for survival analysis is removed. In its place, a corresponding arbitrary action can be added if it causes an even greater increase in quality. If a condition to be removed is found, then the algorithm begins to search

Algorithm 2 Specialisation of the action rule for survival analysis

Input:

$D(A, T, \delta)$ – a data set described by attributes A, observation time T and survival status δ

D_u – a collection of examples that are not yet covered

μ – the minimum number of examples that the new rule must cover

ρ – the maximum rule coverage

τ – indication whether the rule is to be an improving, worsening or any rule

A_{stable} – set of stable attributes from the set A

Output: r – action rule

```
 1: function SPECIALIZE($D, D_u, \mu, \tau, A_{stable}$)
 2:     $\varphi, \varphi_S, \varphi_T \leftarrow \emptyset$        ▷ the premise of the created action rule, a set of source elementary conditions, a set of target elementary conditions
 3:     $W_{ignore} \leftarrow \emptyset$        ▷ a set of elementary conditions already checked
 4:     repeat
 5:         $w_{best_S} \leftarrow$ BESTELEMENTARYCONDITION(D, $D_u$, $\varphi_S$, $\mu$, $\rho$, $W_{ignore}$)
 6:         $W_{ignore} \leftarrow W_{ignore} \cup \{w_{best_S}\}$
 7:         if $w_{best_S} = \emptyset$ then $Continue$
 8:         $a \leftarrow$ ATTRIBUTE($w_{best_S}$)
 9:         $w_{best_T} \leftarrow$ COUNTER-CONDITION($D, \varphi_S \wedge w_{best_S}, \varphi_R, w_{best_S}, \mu, \tau$)
10:         $action \leftarrow$ BUILDACTION($w_{best_S}, w_{best_T}$)
11:         $\varphi \leftarrow \varphi \wedge action$
12:         $\varphi_S \leftarrow \varphi_S \wedge w_{best_S}$
13:         $\varphi_T \leftarrow \varphi_T \wedge w_{best_T}$
14:     until ($w_{best_S} = \emptyset$)
15:     $\hat{S}_S \leftarrow$ KM for set COVEREDEXAMPLES($\varphi_S, D$)
16:     $\hat{S}_T \leftarrow$ KM for set COVEREDEXAMPLES($\varphi_T, D$)
17:     return $r \equiv$ IF $\varphi$ THEN $\hat{S}_S \rightarrow \hat{S}_T$
18: end function
```

anew for the actions left in the premise. The end of rule pruning occurs when no more actions to remove are found, or changing any action to an arbitrary action only degrades the quality of the rule. The end of the procedure can also occur if the rule covers the entire set of examples.

It is important to note that the procedure also checks whether removing a condition or changing it to an arbitrary action will cause the final rule to consist of only arbitrary actions. This is not an interesting case, because such an action rule would not give any information about how the elementary conditions on the left should be changed to get a significant change in the survival curve.

The removal of actions is subject to two restrictions. An action is not removed if its removal would result in the share of all examples covered by a single rule exceeding ρ parameter, which prevents the creation of overly general rules. The second restriction is the maximum share of common examples that the source and target rules can contain at the same time. It is set by ξ parameter. If the value of common examples exceeds ξ value, the action will not be removed, which prevents the creation of actions in which the groups of examples from source and target rules do not significantly differ from each other.

When trimming an action rule, the positions of the survival curves are also checked, because we do not want to change the relative position of the curves after changing or removing an action.

To handle missing values, the so-called ignored values strategy is used. When searching for possible conditions, missing values are ignored and rules are built only based on known observation values. Observations are not included by a rule if they have a missing value for the attribute tested by this rule. The strategy has similar efficiency to more advanced methods of handling missing values [169] and does not require additional steps, which is necessary in the case of imputation methods.

The proposed method combines action mining and survival analysis, offering a unique approach. Previous methods of action mining were not able to handle censored observations. Considering censored data as event-free would skew the model in favor of the event-free outcome. Moreover, when defining action rules for such data, it is essential to include a survival curve in the rule conclusion. As a result of the proposed exploratory analysis, an interpretable model is obtained, which explains the actions based on the data attribute values, leading to a survival curve with a distinct character.

3.5.3 Survival action rules for promotion task

Although the name *survival analysis* is particularly associated with the medical field, the application of the domain is much broader. The event under study of survival analysis could be a relapse or death of a patient, but it could also be an equipment failure or loss of a customer.

In this study, survival analysis and survival action rules are used in context of promotions. Each promotion lasts for a preset number of days. In earlier studies, the promotion was considered as a whole, without dividing it into individual days of the promotion. The proposed indicators were determined for the entire promotion. For example, the value of the indicator AVG. BASKET was determined based on the average of each day of the promotion. In this study, we will go down to a lower level of aggregation and the indicators will be determined separately for each day of the promotion.

We assume that for each indicator an expert could indicate what the minimum expected value of the indicator is – this would be our threshold. The threshold could be set individually for a product or for a whole group of products. In this case, we are able to verify whether or not the promotion has fallen below the preset threshold over the course of its duration. A drop below the threshold will be our event in the survival analysis – a promotion for which there was a drop below the preset threshold will have *survival status* set to 1. In this case, the *survival time* will be the day of the promotion in which the event occurred. For other cases (*survival status* is equal 0), the *survival time* will be set to the number of days of the promotion. These will be our censored observations, since we do not know whether there would have been a drop below the set threshold if the promotion had lasted longer.

In this research, it was assumed that only promotions that:

- had a indicator value above the set threshold throughout the whole promotion period or
- during the promotion, the indicator fell below the set threshold and maintained as this until the end of the promotion

will be considered. In preparing the set for the survival analysis, no consideration was given to promotions for which there was a drop below a set threshold, and then the indicator came back above again.

Such survival information can be used for further analysis. Survival curves can be plotted for groups of products or for individual products, or survival rules can be inducted. We, on the other hand, will perform an analysis of the surviving action rules to determine changes that could be made to reduce the probability of the indicator falling below a set threshold.

Dataset preparation

The research presented here focused on the indicator AVERAGE VALUE OF A BASKET CONTAINING THE PROMOTED PRODUCT BUT DISREGARDING THE VALUE OF THE PROMOTED PRODUCT (shortcut: AVG. BASKET WITHOUT ITEM). It was chosen because it is fairly universal for many products and does not include the price of the promoted product in its value. In this study, we relied on a different dataset than the one presented in previous sections of chapter 3, as we only had daily product sales data for the 2017-2018 promotion. For this reason, the dataset was limited. In addition, as mentioned earlier, the dataset did not include promotions for which the indicator fell below a preset threshold after which it was again above the limit.

The indicator threshold was set jointly for all products included in the same product group. Arbitrarily a threshold was set as a 20-th percentil for values of AVG. BASKET WITHOUT ITEM indicator withing group of products.

In the survival dataset, following conditional attributes were used:

- *weekday*,
- *tv* –logical value describing if promotion was advertised in TV,
- *radio* – logical value describing if promotion was advertised in radio,
- *internet* – logical value describing if promotion was advertised in Internet,
- *spec_placement* – logical value describing if product had special placement in store,
- *prom_diff* – logical value describing if promotion was advertised in a different way,
- *price* – price of a product,
- *pricechange* – change of a price,
- *nb_all_promotions* – number of all promotions in a store,
- *nb_all_promotions_prom_diff* – number of all promotions that were advertised in a different way;
- *nb_all_promotions_tv_radio_or_internet* – number of all promotions that were advertised on TV, radio or internet,

- *nb_all_promotions_tv_radio_internet_or_prom_diff* – number of all promotions that were advertised on TV, radio, internet or in a different way;
- *first_date_day* – number of a day in the month of a first day of promotion,
- *first_date_day_of_year* – number of a day in the year of a first day of promotion,
- *first_date_month* – month number of a first day of promotion,
- *first_date_week* – week number of a first day of promotion,
- *season.*

Case study

The case study of using survival action rules for promotions will be presented for promotions from dairy products group. In this study, we want to create actions that will lower the probability of indicator going below the set threshold. Because of this, the experiment was performed with any assumption that the generated rules should be improving. Thus, the τ parameter value was set to 'BETTER' (see Algorithm 1), which meant that the target rule should conclude with a curve that was above the survival curve of the source rule. The parameter μ specifying the minimum number of examples not yet covered that the new rule must cover was set to 500, the parameter ξ specifying the maximum percentage of examples common to both rules was set to 0.1, and ρ specifying the maximum rule coverage was set to 0.5. The attibutes *first_date_day*, *first_date_day_of_year*, *first_date_month*, *first_date_week* and *season* were set as stable attributes. It means that for these attributes the actions in survival action rules would not be created.

Summary of the results of the survival action rules induction is presented in Table 3.5. It contains selected indicators of the quality of the rule sets. The table includes the percentage of rules for which the p-value of the log-rank test is less than 0.01 ($\%r_{p<0.01}$) without and with adjustment for the scenario in which multiple results were obtained (Bonferroni correction [128] was used). The summary shows that the obtained rule sets are of good quality – after taking into account the correction for p-values the percentage of rules with p-value < 0.01 is 100%.

indicator of the quality of the rule sets	value
number of rules ($\#r$)	10
number of rules without any action in premise	0
total number of conditions ($\aleph_c$)	41
total number of actions ($\aleph_a$)	35
number of “ANY” actions	8
average number of conditions per rule ($\overline{\#c}$)	4.1
average number of actions per rule ($\overline{\#a}$)	3.5
mean % of examples covered by source rule	25.7
mean % of examples covered by target rule	38.9
mean % of examples covered by both rules	0.01
min # of conditions/actions in a rule ($\mu_{c/a}$)	2/1
max # of conditions/actions in a rule ($M_{c/a}$)	7/6
% of rules with p-value < 0.01 ($\%r_{p<0.01}$)	100
% of rules with p-value < 0.01 after correction	100
% of rules with p-value < 0.05 ($\%r_{p<0.05}$)	100
% of rules with p-value < 0.05 after correction	100

Table 3.5: Evaluation indices of the set of survival action rules generated for the promotions for dairy product group.

The occurrence frequency of each attribute in the generated rules can be analysed from Table 3.6. The *nb_all_promotions* attribute occurs in 8 of 10 rules generated for the three defined settings. It means that only 2 action rules did not include any condition with this attribute. Therefore, it can be concluded that it is the most important attribute that changes the form of the survival curve.

Attribute	Number of occurrences
nb_all_promotions	8
nb_all_promotions_tv_radio_or_internet	6
price	6
weekday	4
first_date_day_of_year	3
nb_all_promotions_prom_diff	3
nb_all_promotions_tv_radio_internet_or_prom_diff	3
pricechange	3
first_date_day	2
first_date_month	1
radio	1
tv	1

Table 3.6: Ranking of attributes according to the number of occurrences in rule premises for the promotion dataset.

The premise parts of the generated rules are presented in Table 3.7. The survival curves from the conclusions of the selected rules are presented in Fig. 3.28. In accordance with the configuration adopted the selected rules are improving ones, thus the curves for the target parts are above the curves for the source parts.

Id	Premise
r1	*(nb_all_promotions_tv_radio_or_internet, (-inf, 13.5) -> <0.5, inf)) AND (nb_all_promotions_prom_diff, (-inf, 43.5) -> <24.5, inf)) AND (price, <0.59, inf) -> <0.97, inf)) AND (nb_all_promotions, <36.5, 77.5) -> <145.5, inf)) AND (pricechange, <-0.363, inf) -> <-0.3345, inf))*
r2	*(nb_all_promotions_tv_radio_or_internet, <15.5, inf) -> <51.5, inf)) AND (first_date_day_of_year, (-inf, 333.5)) AND (nb_all_promotions, (-inf, 898.0) -> <272.5, inf)) AND (first_date_day, <21.5, inf))*
r3	*(nb_all_promotions_tv_radio_internet_or_prom_diff, <52.5, inf) -> <134.5, inf)) AND (nb_all_promotions_tv_radio_or_internet, <15.5, inf) -> <51.5, inf))*
r4	*(nb_all_promotions_tv_radio_internet_or_prom_diff, (-inf, 76.5) -> <57.5, inf)) AND (nb_all_promotions_tv_radio_or_internet, (-inf, 18.0) -> ANY) AND (nb_all_promotions_prom_diff, (-inf, 61.5) -> <45.0, inf)) AND (first_date_day_of_year, <20.5, 339.0)) AND (price, (-inf, 1.57) -> ANY) AND (nb_all_promotions, <36.5, 223.5) -> <205.5, 489.5)) AND (pricechange, <-0.3175, inf) -> <-0.2935, -0.093))*
r5	*(nb_all_promotions_tv_radio_or_internet, (-inf, 18.0) -> <2.5, inf)) AND (nb_all_promotions_prom_diff, (-inf, 117.5) -> <22.5, inf)) AND (price, (-inf, 1.57) -> <1.6400000000000001, inf)) AND (first_date_day_of_year, (-inf, 339.0)) AND (nb_all_promotions, <36.5, 301.5) -> <143.5, inf))*
r6	*(nb_all_promotions_tv_radio_internet_or_prom_diff, (-inf, 131.5) -> ANY) AND (price, (-inf, 2.54) -> <0.72, inf)) AND (weekday, (-inf, 2.5) -> <3.5, inf)) AND (nb_all_promotions, <92.5, inf) -> <199.5, inf))*
r7	*(tv, {0} -> ANY) AND (price, (-inf, 3.34) -> ANY) AND (weekday, (-inf, 2.5) -> <3.5, inf)) AND (nb_all_promotions, <184.0, inf) -> <200.5, inf)) AND (pricechange, (-inf, -0.07899999999999999) -> ANY)*
r8	*(price, (-inf, 3.52) -> ANY) AND (weekday, (-inf, 1.5) -> <3.5, 4.5)) AND (nb_all_promotions, <130.0, inf) -> <199.5, inf)) AND (radio, {0} -> ANY)*
r9	*(weekday, (-inf, 2.5) -> <3.5, inf)) AND (nb_all_promotions, <135.5, 389.5) -> <192.5, inf))*
r10	*(nb_all_promotions_tv_radio_or_internet, (-inf, 14.5) -> <18.5, inf)) AND (first_date_month, (-inf, 11.5)) AND (first_date_day, (-inf, 29.5))*

Table 3.7: Survival action rules generated for the promotions.

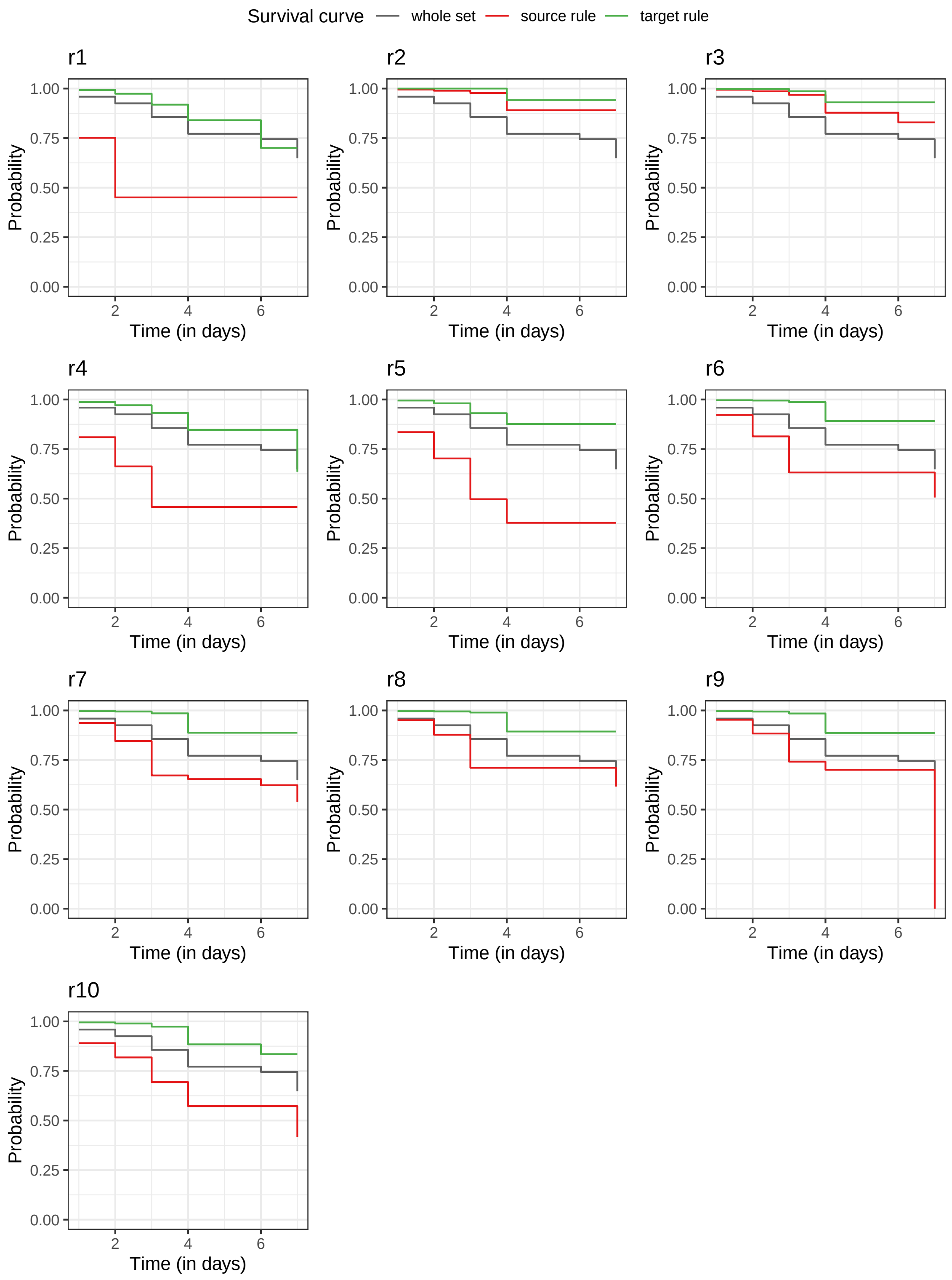


Figure 3.28: Survival curves of rule conclusions from Table 3.7. The curve corresponding to the source rule is in red, the curve corresponding to the target rule is in green.

Analyzing rules in Table 3.7 it can be seen that the number of conditions and the number of actions in the premise of the generated rules reaches a maximum of 7. The number of all attributes (not including variables denoting time and survival status) in the prepared dataset is 17. This means that the proposed method did not generate overly specific rules containing conditions for all attributes. From Table 3.5 it can be concluded that all of the generated rules have low p-value of the log-rank test between the source and target rule curves. This means that for each rule, the source curve and the target curve are significantly different from each other.

Further, more detailed analyses, may include inspection of a specific rule. To provide an example of the analyses, the r5 rule was chosen, the form of which is shown in Table 3.7 and the survival curves in Figure 3.28.

The premis of rule r5 consist of 5 conditions including four actions and one elementary condition (stable attribute). Actions were inducted for attributes:

- *nb_all_promotions_tv_radio_or_internet*,
- *nb_all_promotions_prom_diff*,
- *price*,
- *nb_all_promotions*.

Density histograms presented in Figure 3.29 were generated for these attributes. The histograms were created for the source and target conditions using examples covered by the source and target rules. The histogram was not generated for condition *first_date_day_of_year*, which also is present in rule r5, because it is a stable attribute.

From density histograms presented in Figure 3.29 it can be concluded that the rule indicates to increase the number of promotions taking place in the store in order to reduce the probability that during the promotion the value of the basket will fall below the set threshold. Such a rule seems to be in line with intuition – the more promotions, the customer may want to buy more products from the promotions, which affects the value of the basket,

An action related to the price increase also appeared in the rule. The density histogram presented in Figure 3.29 fo this attribute points out a significant change in the average price for promotions included in the target rule comparing to to the source rule. This change results in a higher survival curve. It is speculated that this may be because a big promotion will cause customers to come to the store just to buy a product at a bargain price. Such behaviour may result in few goods in the shopping basket. In the case when only the promoted product is bought, the value of the examined indicator will be 0, because the indicator determines the value of the shopping basket without the product to which the examined promotion applies.

A stable attribute *first_date_day_of_year* also appeared in the examined rule. Its value was specified as smaller than 339. That is, the rule applies to promotions that take place from the beginning of the year to December 5. December 6 is Santa Claus Day, which is often celebrated in Polish, especially among children. It is possible that after this date the period of shopping related to the future Christmas holidays begins – Christmas gifts may be bought, and in the days closer to Christmas Eve a lot of shopping related to this holiday will already be done. Christmas is celebrated in most Polish homes and is often associated with large purchases, which will also affect the value of the examined indicator. It is possible that the studied rule will not apply during this particular period. However, it should be noted that the given data set contained data only from 2017-2018, so it may be too small number of data to draw such conclusions.

The presented case study shows how the innovative separate-and-conquer survival action rule learning algorithm can be applied to the analysis of promotions. Such results show what changes (actions) could be made to change indicator values and extend the time for which a promotion meets a set acceptable indicator value. The performed analysis showed that the generated rules allow to indicate the conditions that influence the change of the KM estimator value. The good quality of the generated rules is confirmed by low p-value of the log-rank test for the source and target rules. Based on this, it can be concluded that the proposed algorithm for the induction of survival action rules for examined case gave satisfactory results. Further research could look at possibilities of creating survival action rule for multiple indicators simultaneously.

3.6 Conclusions

The given chapter presents ways to forecast and analyze the effectiveness of promotions. Firstly, 6 indicators are proposed to evaluate the effectiveness of promotion. Then the preparation of data sets that can be used for such analysis is presented. The process of preparing conditional attributes and selecting data is outlined. Next, studies of various methods for creating predictive models of indicator data for whole groups of products were performed. Then similar studies were carried out for individ-

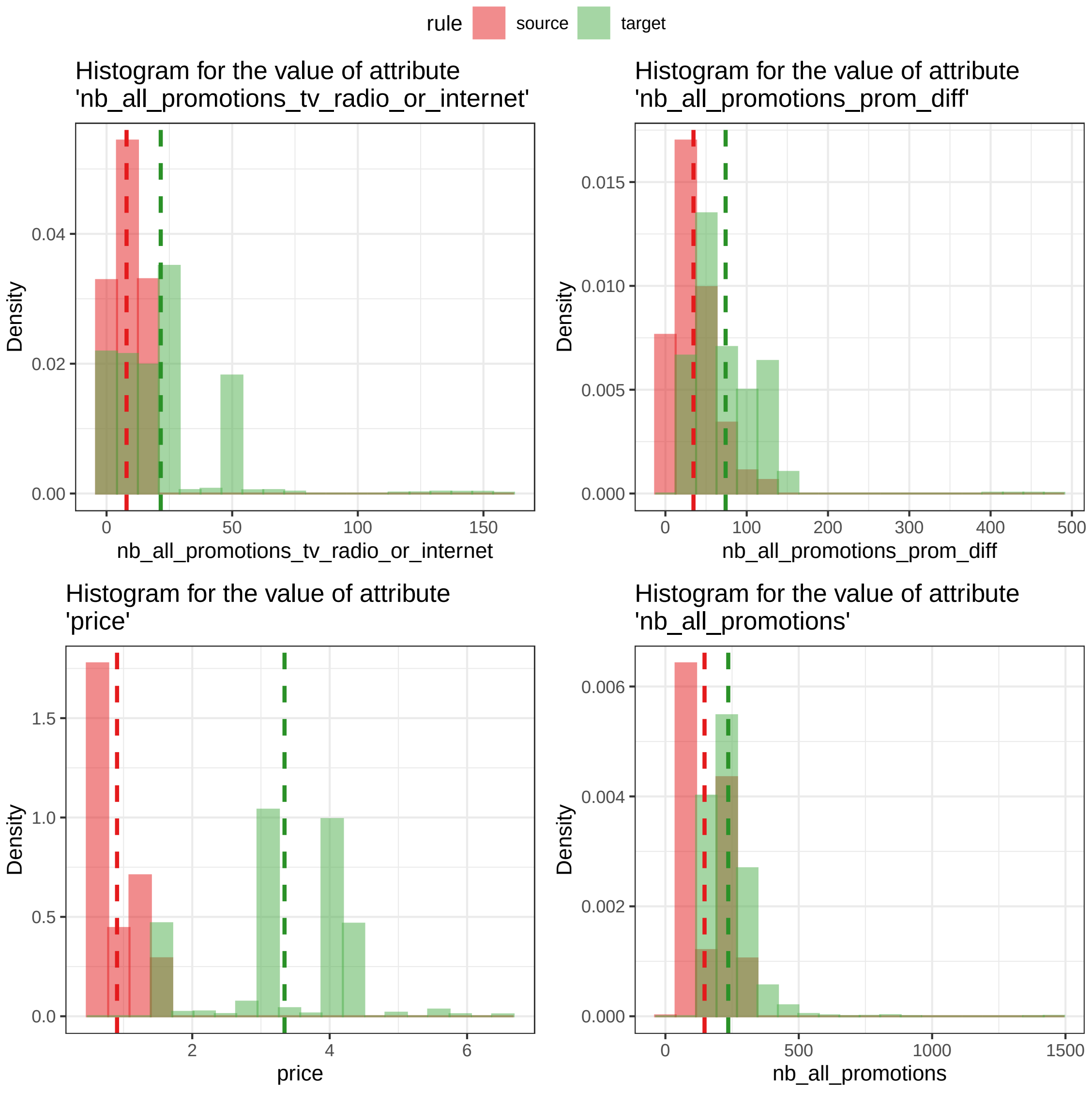


Figure 3.29: Histogram of the density of attribute values occurring in the actions in the r5 rule. Histograms are drawn for the source and target conditions on the examples covered by the source/target rule. The dashed line indicates the mean value.

ual products. The research shows that it is better to create such models for groups of products, since only a small number of products have enough data to prepare such models. There may also be products for which there was almost no promotion at all, and at such a time the possibility of using a predictive model is beneficial. The study also found that the XGBoost model performed best among the approaches studied. The next part of the research presents the practical use of predictive models for promotional performance indicators built on XGBoost. The possibility of using them in a price change simulator and in a recommendation system for the best promotion is shown. Finally, an innovative approach of using the proposed induction algorithm for survival action rules to analyze promotions is presented. The results of such an analysis give information on what changes (actions) could be applied to make the promotion meet the expected value of the indicator for the longer time period of the promotion.

4 Top-down forecasting

Product stocking can be carried out in a variety of ways. At times, products are ordered on an ongoing basis, based on current consumption. Other times, orders are placed well in advance, in which case forecasting future demand becomes critical. This often occurs, for instance, when a store signs a contract with an ice cream supplier for the summer season. A similar approach is used in the fashion industry. However, there are some key differences.

The first challenge lies in the seasonal nature of fashion sales (i.e., spring/summer and autumn/winter sales). Another challenge is the difficulty in capturing and interpreting information about intangible product attributes such as fashion and emotions. Additionally, it's essential to forecast demand over a long time horizon in this business, given that many goods are imported from countries where they are extensively produced. Ordering products in advance is necessary to ensure timely production, shipping, marketing, and distribution across international borders.

On the other hand, fashion merchandise evolves rapidly. Products are rarely offered for multiple seasons, and the majority are only sold during one sales season, resulting in a short sales history. This applies to both natural seasonal shifts and ever-changing trends that can differ greatly. Another factor to consider is the relative rarity of sales of a given product model in a specific size and color at a particular store. In the fashion industry, products are often sold in collections, and new products typically replace most of the previous season's offerings.

Given the need to forecast sales far in advance for products that have yet to be released, it's evident that the forecasting process becomes more complex. An expert may assist in predicting future sales of new product lines, but forecasting sales for each week of the upcoming season for a specific product in a particular size and color poses a challenge. While an expert can use their domain knowledge to forecast sales for a group of products, it's challenging to predict sales for each specific product in a particular size and color. As a result, we propose our approach to forecasting demand for such circumstances.

The research was carried out as part of the project co-financed by European Funds entitled 'Decision Support and Knowledge Management System for the Retail Trade Industry (SensAI)' (POIR.01.01.01-00-0871/17-00).

4.1 Problem statement

In the research, that was also described in paper [74], we were dealing with the problem of an estimation of the conditional probability of sales of fashion goods, taking into account certain nominal characteristics of these goods and taking into account the time factor. Our goal was to find a way to obtain a forecasts for a specific product based on predictions made for a product categories, where specific product is a product described by colour, size and product type. This is a case of top-down forecasting. As an example, our forecast should give prediction not only for blouse XYZ but for blouse XYZ in size M and in colour green. Having the forecast for group of products (e.g. for specific group of blouses), the problem boils down to dividing the forecast into smaller forecasts, that is, determining the probability (fraction) by which the overall forecast will be multiplied.

The forecasts for a certain product category were provided to us by a business partner and are proprietary. However, we can conclude that forecasting for the category is feasible, given its past sales history. Furthermore, since the number of product categories is limited, an expert with domain knowledge can make accurate forecasts for the categories of products.

In our research we studied different approaches to making demand forecast for the specific products from fashion retail based on higher level forecasts. We used naive method, custom nearest neighbour approach, parametric linear mixed model and an ensemble approach.

We had to create predictions for weekly aggregates for 29 weeks ahead. It was a problem of long-term predictions.

4.1.1 Datasets

We were working with real life data describing sale of fashion products. The dataset contains data from 2016 to the end of 2021, so it partially cover the period of COVID-19 pandemic and lock-downs. The data from April 2021 to November 2021 was our test dataset, because for this period we had from our business partner not only predictions for the whole category, but also their predictions for unique products and we could treat it as our baseline.

In the process of data preparation we created dataset in which each observation describe sale of one specific product in one week. The attributes that we used were as follow:

- ID attributes: product name, category name, date of a first day from a considered week.
- Attributes connected with a product in sale: product type, size, main colour and colour undertone.
- Attributes connected with time of a sale (describing the week of an observation): the week number, season, quarter of the year, the number of days until the closest holiday or event (such as Christmas, Easter, a national holiday, Valentine's Day, etc.), the number of days since the closest holiday or event, the number of holidays or events that occurred during the considered week, the number of holidays or events that occurred during three weeks — the week under consideration, the prior week, and the following week, binary feature that described if it is the last week of a year.
- Sales-related characteristics: the quantity of units sold, the number of units sold within a category, and the percentage of sales in a particular category that correspond to the sale of the product in question.
- Attributes connected with trend and seasonality: we used Prophet tool [2, 149] to decompose data from train dataset into trend and seasonality for each product. Results obtained were applied to the test samples.

Because of making predictions for long-horizon we couldn't use data that may be not known in a future as weather or sale from weeks just before the forecast week. In our dataset, we didn't have information about price or planned promotions. Forecasts were made for all stores collectively, with no breakdown of sales at individual stores.

The training set included almost 62,000 observations from January 2016 to April 2021. The test set included more than 9000 observations from April 2021 to November 2021.

In the table 4.1, we present number of product types, colours and sizes in each category.

Category	Number of product types	Number of sizes	Number of colours
A	1	12	3
B	3	4	3
C	4	34	4
D	1	4	1
E	2	2	1
F	2	4	3
G	1	5	3

Table 4.1: Number of products in each product type, size and colour in each category of products in analyzed dataset.

4.2 Proposed approaches

4.2.1 Naive approach (data cube approach)

Initially, we employed a naive method to partition the sales forecast for a product category into individual forecasts for unique products within that category. This involved determining a probability (weight) that the category forecast should be multiplied by to arrive at the forecast for a unique product, with the probability ranging between 0 and 1.

For a given time period, this approach required the generation of a 'cube' based on historical data, with each side of the cube representing an attribute by which the product can be characterized. The intersection of these dimensions resulted in a weight (fraction) that was used to multiply the sales forecast for the product category to which the product belongs. The outcome of this computation was a sales forecast for a specific product at a particular point in the future.

Although the term "cube" is used in this description, it should be noted that the number of attributes used to characterize the product may be more or less than three. However, in our data, the specific product was characterised by three attributes: colour, size and product type.

We can generate multiple "cubes" over time, as presented in Figure 4.1 because cube values can be calculated based on other data as they come in. This means that if forecasts are made for each week, the probabilities (weights) described by the cubes will change depending on what the sales of a particular product were in that week.

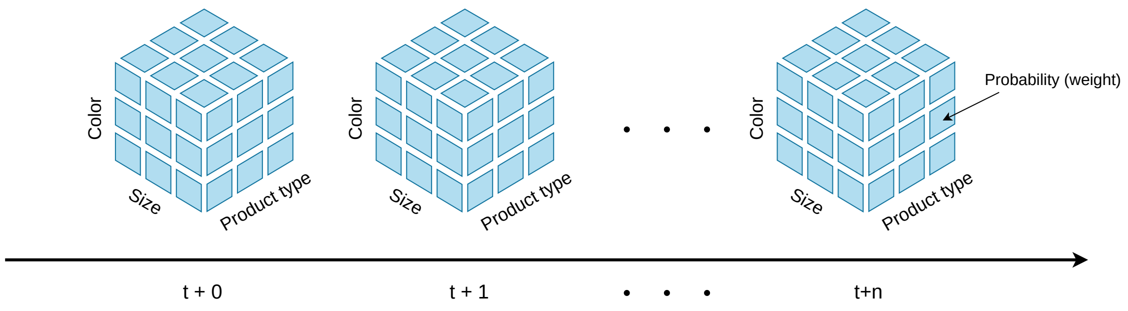


Figure 4.1: Naive approach - multiple 'cubes' over time.

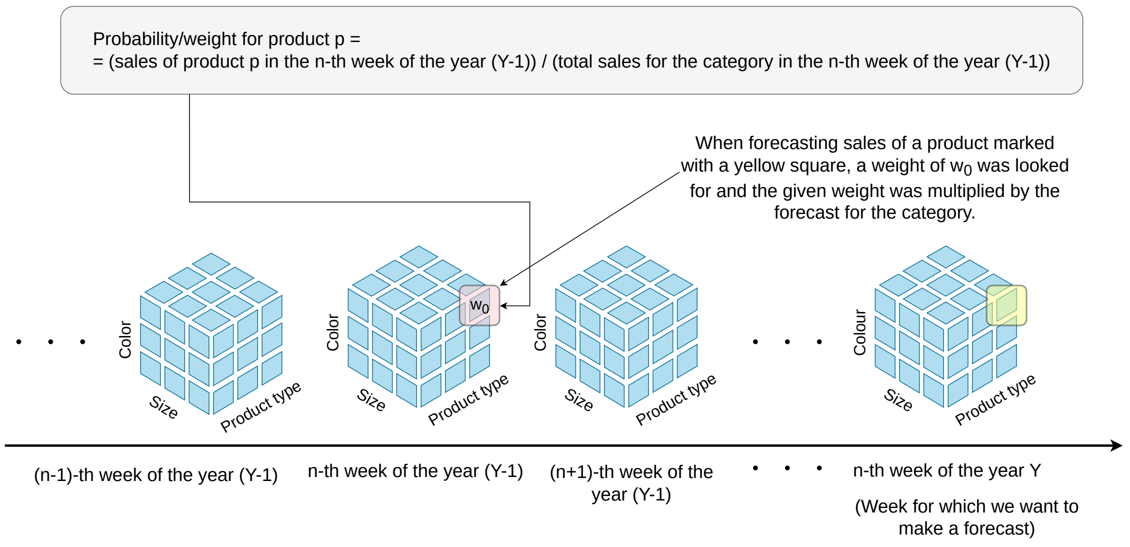

Probability/weight for product p =
= (sales of product p in the n-th week of the year (Y-1)) / (total sales for the category in the n-th week of the year (Y-1))
When forecasting sales of a product marked with a yellow square, a weight of w_0 was looked for and the given weight was multiplied by the forecast for the category.
w_0
Color
Colour
Size
Product type
(n-1)-th week of the year (Y-1)
n-th week of the year (Y-1)
(n+1)-th week of the year (Y-1)
n-th week of the year Y
(Week for which we want to make a forecast)

Figure 4.2: Simple naive approach - calculating final forecast.

The proposed simple naive approach assumes that a forecast needs to be made for a product labeled p in category c for a given week n in a particular year Y. In this approach, the sales figures for the same week in the previous year $Y-1$ are analyzed. The sales of the product p during that week are then divided by the total sales in category c during the same week to obtain a probability value. This probability value is used as a weight to multiply the forecasted sales figure for category c in week n of year Y. The resulting figure is considered the forecasted sales for the product p. The process of calculating final forecast for product p is presented in Figure 4.2.

The extension of the suggested method is to calculate the weight based not only on the previous week, but also on the week before that and the week after that. This ensures that the weights are averaged and minimizes the impact of outliers on the resulting forecast. On the basis of Equation 4.1, it would be determined.

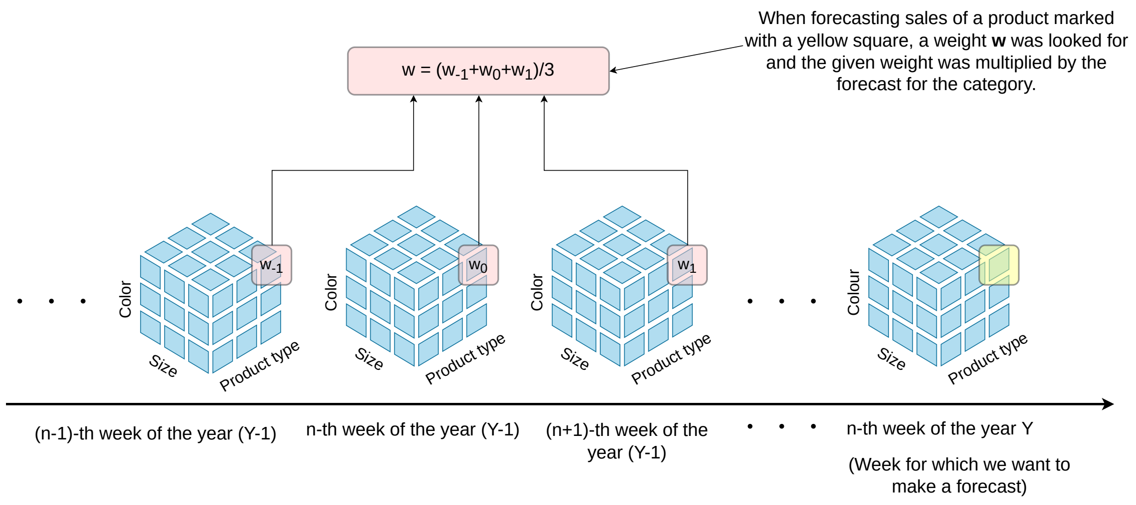

When forecasting sales of a product marked with a yellow square, a weight **w** was looked for and the given weight was multiplied by the forecast for the category.
w = (w-1+w0+w1)/3
w-1
w0
w1
Color
Colour
Size
Product type
(n-1)-th week of the year (Y-1)
n-th week of the year (Y-1)
(n+1)-th week of the year (Y-1)
n-th week of the year Y
(Week for which we want to make a forecast)

Figure 4.3: Extended naive approach - calculating final forecast.

$$y_{p,n,Y} = \frac{1}{3} \sum_{i=-1}^{1} \frac{s_{p,n+i,Y-1}}{s_{c,n+i,Y-1}} y_{c,n,Y} \tag{4.1}$$

where s means real sale, y is a forecast, p is a product, for which we calculate the forecast, c is a category of a product, n is a week number, Y is a year.

The Figure 4.3 presents the way of calculating final forecast in extended naive cube approach.

The naive model assumes that no expert knowledge is used, that is, no reference products or models are established for products. Only the knowledge contained in the database is used, that is, we have information about the characteristics of the product (product type, color, size) and what category the product belongs to. It can big an advantage of this approach. Another advantage of the naive approach is the lack of model training and the absence of the problem of overfitting. The disadvantage of this solution is the fact that the weight is naively searched for one year earlier or around a given date. The solution does not take into account the changing trend, nor does it consider holidays and special days. For this reason, other approaches, which would be based on assumptions of weight averaging, were investigated.

4.2.2 Nearest neighbour (KNN)

We suggest a nearest neighbour (KNN) strategy as a more complex forecasting method. Although this is a non-parametric approach that is frequently used in classification tasks, we used it to identify comparable observations from past data.

Given an input vector, this method determines the distance (using a selected distance measure) to observations from a training dataset. Its closest neighbour is the single observation whose conditional attribute vector resembles the new feature vector the most. The KNN method was chosen as an extension of the naive method, which made a forecast using sales data from the previous year. The KNN algorithm makes a forecast of the weight (sales of a given product within an entire category) by searching for the most similar observations from the entire range of historical data. The resulting weight is multiplied by forecast for the entire category, resulting in a sales forecast for a specific product.

In this approach, an extended dataset was used. The dataset has additional attributes describing the week, number of events or season. In order to use some categorical attributes, additional input data had to be provided. Additional data is needed for the size attribute. The analyzed collection contained data for various products, including shirts, socks, and bras. Each of these products have different sizes. For example, shirts may have sizes XS, S, M, L, XL or 36, 38, 40, etc., socks are labeled 36/38, 38/40, etc., and bras have sizes such as 70B, 70C, 75B, 80D, 90C. To calculate the distances between different observations using the size attribute, the original sizes were converted into numerical values based on the historical data. We divided the historical dataset by size — a separate subset for each category of products, which have different clothing sizes. Then, for each of the sizes in the set, we determined the percentile values within those subsets. That is, if we forecasted sales for socks in size 36/38, we converted size 36/38 into the numerical value — percentile that this size represented in the historical data.

The attribute “product type” also received additional entry data. This attribute had nominal values, which served as identifiers for the various fashion styles and had no further significance. We suggested using *product types dissimilarity table*, which shows distance between different product types. If we calculated the distance between products that had the same product type, the distance was equivalent to 0. If we measured the distance between two products that belonged to the same clothing category but were of different product types, for example the distance was equal to 0.5. If we calculated the distance between goods that belonged to various clothing categories, the distance was equal to 1.

In the KNN approach we used following attributes:

- week number,

- product type
- size,
- main color,
- secondary color,
- season,
- quarter,
- number of days to the nearest event,
- number of days since the nearest event,
- number of events/holidays that occurred in the considered week,
- number of events/holidays that occurred during the considered week, in the previous week, and in the following week

Then, normalization of numerical variables was applied using the Min-Max method. The numerical variables were normalized to the range from 0 to 1.

The schema of the proposed procedure is presented in Figure 4.4. The KNN method is called with the parameter k=3. This parameter value was selected from the set of values k={1,2,3}, as the value giving the best model results.

Let's assume that we want to predict the demand of the product XYZ in week starting from 2021-05-17. We will call this observation as a o_1. Assume also that the product XYZ is of product type "X", has colour "Y" and size "Z". Firstly, we need to prepare a training set of historical data, among which we want to search for the nearest neighbour for the observation o_1. The training dataset is limited to historical data of the specific product XYZ only when we have at least one year of historical data for this product. The training dataset is limited to historical data of all products of product type "X" only when at least one product from this group has at least one year of historical data. If these assumptions are not met, our training set is a dataset of historical data of reference products to product XYZ. These are all products whose product types have a distance of 0.5 to XYZ in the *product types dissimilarity table*. After obtaining a new train dataset, the algorithm KNN finds the nearest neighbour to the observation o_1 – we will call this nearest neighbour *onn*. Then we can calculate the weight of *onn*. The weight is a ratio of the total

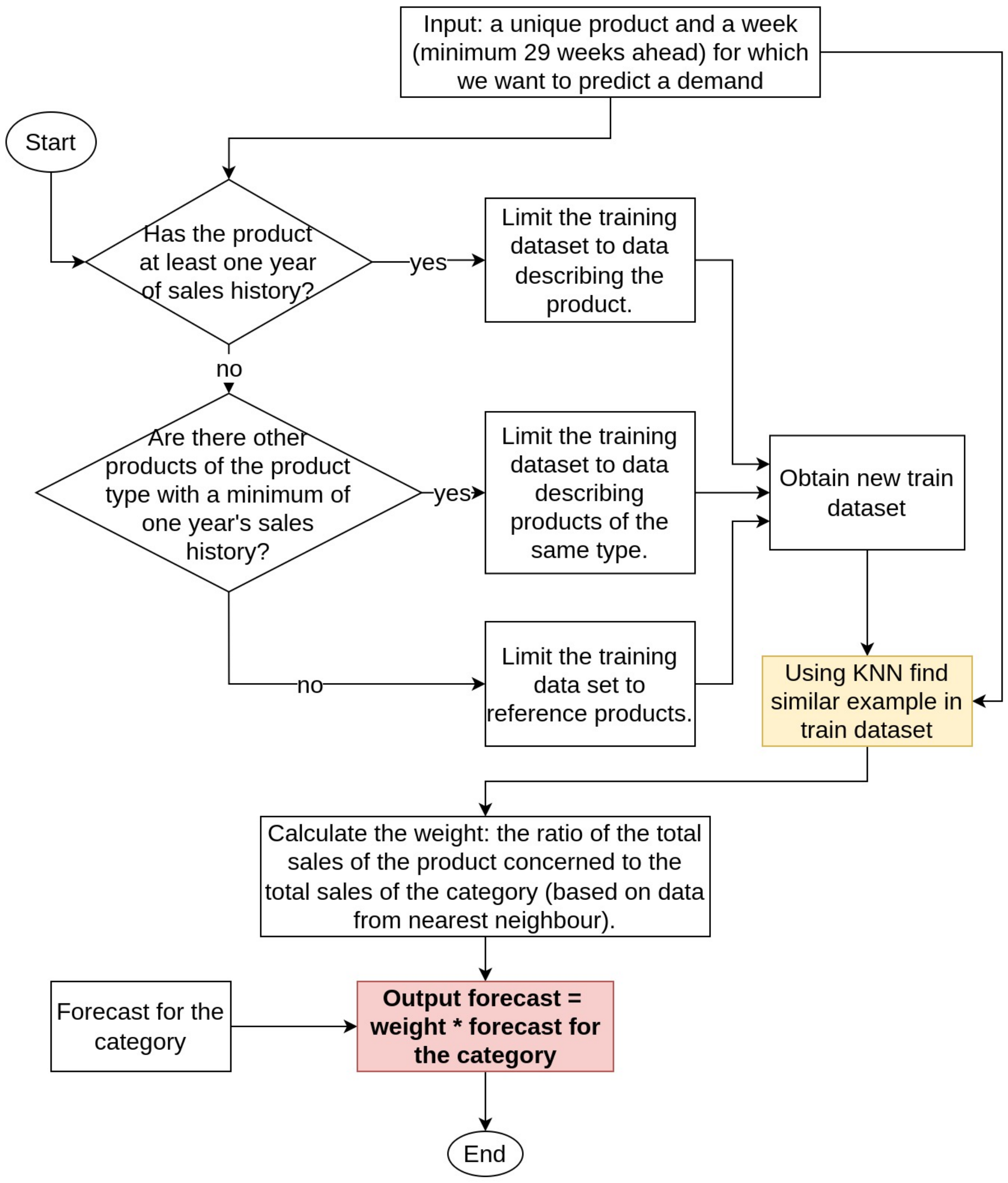

Input: a unique product and a week (minimum 29 weeks ahead) for which we want to predict a demand
Start
Has the product at least one year of sales history?
yes
Limit the training dataset to data describing the product.
no
Are there other products of the product type with a minimum of one year's sales history?
yes
Limit the training dataset to data describing products of the same type.
Obtain new train dataset
no
Limit the training data set to reference products.
Using KNN find similar example in train dataset
Calculate the weight: the ratio of the total sales of the product concerned to the total sales of the category (based on data from nearest neighbour).
Forecast for the category
Output forecast = weight * forecast for the category
End

Figure 4.4: The schema of the custom nearest neighbour procedure for obtaining the forecasted demand of a product.

sales of the product in o_{nn} to the total sales of the category, to which the product from o_{nn} belongs. So for example, if the *onn* describes the sale of a product ABC in week starting from 2020-07-06, the weight will be the sale of a product ABC in week 2020-07-06 divided by the sale of a whole category, to which the product ABC belongs to, in the same week. This weight is multiplied by the forecast for a category, to which the product XYZ belongs to, and for the week starting from 2021-05-17. As a result, we obtain a prediction for the observation o_1.

It is worth noting that the KNN method relies on a similar assumption as our naive approach — the weight is determined as a proportion of the product's sales to the sales in the entire category. However, in the KNN method, the weight is not necessarily searched for a week that occurred exactly one year earlier. Additionally, the nearest neighbor, which is used to determine the weight, may refer to historical data for a different product than the one being forecasted.

Next, we tried to optimize the range of conditional attributes and the number of neighbors. For this reason, a validation set was extracted from the training set - these were data from the last 29 weeks from the training set.

Experiments were conducted for values of K=[1..8]. Optimization of the number of features used in the KNN model was done using climbing strategy. Initially, the KNN model was taught using all the proposed features. Then one feature at a time was temporarily removed and the effect of the removal on the MAE value was checked. Finally, in a given iteration, the feature that caused the greatest decrease in MAE value on the validation set was removed. There could also be a feature removed that did not cause a change in the MAE value. Experiments were conducted until all subsequent deletions caused an increase in the MAE error on the validation set. This experiment was performed for each value of K. In the process of climbing backwards, the characteristics describing the product - product type, color, size - were not involved. Attribute describing week number couldn't also be removed in the optimization process.

4.2.3 Linear mixed model (LMM)

Linear mixed models are an extension of simple linear regression models and can be used for data with a hierarchical structure which is observed in fashion. These models incorporate fixed and random effects:

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{Z}\boldsymbol{u} + \boldsymbol{\varepsilon} \tag{4.2}$$

where $\boldsymbol{y}$ is a vector of outcome variable, $\boldsymbol{X}$ is a matrix of predictors, $\boldsymbol{\beta}$ is a vector of fixed-effects regression coefficient, $\boldsymbol{Z}$ is a design matrix for random effects, $\boldsymbol{u}$ is a vector of random effects and $\boldsymbol{\varepsilon}$ is a vector of residuals [57].

In this approach, a model for each product group was estimated with random effects defined by product, size, and color. In groups where there was only one product, random effects were estimated only for size and color. Due to strong asymmetry of outcome variable (forecast weight) Box-Cox transformation [**?**] was applied.

Using ANOVA, we confirm that there are substantial differences between groups based on product, size, and colour and the outcome variable. The permutation test was used to prove the significance of random effects.

The following characteristics were used as explanatory variables:

- Random effects:
  - product type,
  - color,
  - size
- Fixed effects:
  - season,
  - quarter,
  - number of days until the next event,
  - number of days since the last event,
  - number of events/holidays that took place during the week under consideration,
  - number of events/holidays that took place during the week under consideration, the previous week and the following week,
  - trend and seasonality determined by the Prophet tool.

For linear mixed model optimization was also performed. In the LMM method, only the range of conditional attributes was optimized. Models containing every possible combination of conditional attributes were tested within each category. The optimal arrangement of variables was determined as the one for which the minimum MAE value was obtained on the validation set.

Product ID	Baseline	Simple naive approach	Extended naive approach	LMM without optimization	KNN without optimization	LMM with optimization	KNN with optimization	Ensemble
1111	6.033	6.200	5.600	5.133	5.033	5.133	5.167	5.150
1112	5.933	7.033	6.500	5.567	6.700	5.567	6.400	5.900
1113	5.600	6.700	6.500	5.200	7.467	5.167	5.967	5.567
1114	5.733	5.567	5.167	4.833	7.400	4.867	5.000	4.930

Table 4.2: Example of evaluation of methods by means of ranks – MAE values for 4 example products and for all the approaches.

4.2.4 Ensemble approach

We combined the forecasts of the parametric linear mixed model and the non-parametric KNN technique in the ensemble approach. The forecast returned was the average of forecasts provided by these two models.

4.3 Comparing the proposed approaches

In the next iteration of the study, we performed a summary of the results using ranks. The approach used here was to assign ranks to each method. An MAE measure was calculated separatelly for each product, which is described by the product type, color and size, and for each method tested. Each method was then assigned a rank within the subset - rank 1 was assigned to the method that obtained the smallest MAE error (i.e., was the best in this context), rank 2 was assigned to the second-best method, and so on. It could have happened that two or more methods received the same MAE score. In such a case, all of these methods received an average rank value (e.g., two methods had the same MAE score and were assigned a rank of 2 and a rank of 3, so in the end they both received a rank of 2.5).

An example of rank assignment is shown in Tables 4.2 and in 4.3. Firstly, MAE errors were calculated within each product (Table 4.2). In the next step, based on the determined MAE measures for each product, an appropriate rank was assigned to each method within the product. This step is shown in the table 4.3.

The average rank was then determined taking into account all the products considered (not separately within each category). The results are presented in Table 4.4.

Product ID	Baseline	Simple naive approach	Extended naive approach	LMM without optimization	KNN without optimization	LMM with optimization	KNN with optimization	Ensemble
1111	7	8	6	2.5	1	2.5	5	4
1112	4	8	6	1.5	7	1.5	5	3
1113	4	7	6	2	8	1	5	3
1114	7	6	5	1	8	2	4	3

Table 4.3: Example of evaluation of methods by means of ranks – assigned ranks for 4 example products and for all the approaches based on MAE values from Table 4.2.

	LMM without optimization	LMM with optimization	KNN with optimization	Ensemble	Baseline	Extended naive approach	KNN without optimization	Simple naive approach
Avg. rank	3.526	3.659	3.806	3.848	4.131	5.136	5.482	6.413

Table 4.4: Average ranks for each approach evaluated in stocking problem.

Calculation of statistical significance

We checked the differences between prediction errors for statistical significance using statistical tests. First, a nonparametric Friedman test [59] was conducted, which is used to compare the results of more than two algorithms with each other. This test tests the null hypothesis that all algorithms are equal. After rejecting this hypothesis, a post hoc analysis should be performed using the Nemenyi test [118]. The results of this analysis can be clearly presented using Critical Difference diagrams, which are abbreviated as CD-diagrams. This diagram shows groups of algorithms that show statistically significant differences in the ranks determined by the Nemenyi procedure. CD-diagrams are described in the paper [51]. The Friedman test compares the average ranks of each algorithm. The null hypothesis is that all algorithms are the same, which means that their ranks should be equal. The Friedman statistic is calculated using the formula 4.3.

$$\chi_F^2 = \frac{12N}{k(k+1)}\left[\sum_{j=1}^{k} R_j^2 - \frac{k(k+1)^2}{4}\right] \tag{4.3}$$

where N is the number of datasets tested (in our case, it is the number of subsets of the test set, where each subset refers to one product, so N is the number of unique products), k is the number of methods/algorithms tested, R_j is the average rank of

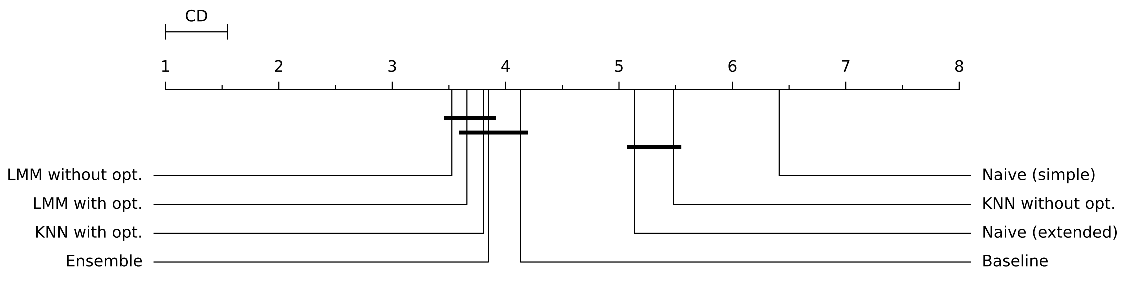


Figure 4.5: Critical Difference diagram for obtained results for each tested method of forecasting for stocking products.

the j-th algorithm. Sometimes the statistic proposed by authors of [82], who found Friedman's statistic too conservative, is used.

In our study, the p-value (p-value) for Friedman's test was smaller than $2.2e-16$. This value is less than 0.05 and for this reason the null hypothesis can be rejected, and this means that there is a statistically significant difference between the methods.

In the next step, we wanted to see which of the tested algorithms are statistically different from each other. For this reason, we drew CD diagrams. This diagram is a form of visualization of the results of the Nemenyi test. When calculating the critical difference value, the alpha value was set to 0.1. The CD diagram drawn for the tested methods is presented in Figure 4.5.

In the upper left corner, the critical difference (CD) value is presented. In our study, CD = 0.548. The average ranks for each method are marked on the axis. The best methods are on the left side of the diagram. If the algorithm took an average rank of 1, it would mean that in each subset (for each product) it obtained the best MAE score. The bold lines in the CD diagram connect algorithms whose average rank does not differ by more than the CD value, that is, it connects algorithms whose results do not differ significantly.

In our study, it can be seen that the LMM method without optimization, LMM with optimization, KNN with optimization and Ensemble approach were combined into one group, then the LMM method with optimization, KNN with optimization, Ensemble approach and the baseline were combined into another group of algorithms, and the third group of methods are the extended naive method and KNN without optimization. The CD diagram shows that there is a statistically significant difference between the average ranks of the LMM method without optimization and our baseline, and the LMM method without optimization gives better results considering all products.

A Wilcoxon test for pairs of observations was still performed to see if there was a significant difference between the MAE obtained for each product for the LMM method without optimization and baseline predictions. This test yielded a p-value = 0.05011. The Wilcoxon test does not show significant differences at the 0.05 level, which is typical, but it does show significant differences at the 0.06 level.

To summarize, the research on how to divide category forecast into product-specific forecasts was based on several proposed approaches: the naive approach in two versions, LMM model with and without optimization, custom KNN model with and without optimization, and Ensemble approach. The study was carried out for a long horizon (29 weeks), which was intended to simulate the product ordering horizon and thus correspond to the problem of stocking products. The best results were obtained for LMM method without and with optimization, KNN with optimization and Ensemble approach. The ranking of methods indicates that considering all products, it is the LMM method without optimization that gives the best results. Statistical tests indicate that there is a significant difference between the LMM method without optimization and the baseline. The CD diagram indicates that such a difference is at the 0.1 level, while the Wilcoxon test shows a significant difference, but at the 0.06 level (p-value = 0.05011). Both versions of the naive method and the KNN method without optimization perform significantly worse than the other methods tested. It can suggest that in this problem naive method, that looks for benchmark in data from one year before, is too simple and more advanced Machine Learning methods should be incorporated to this problem.

In the research, we presented how those responsible for ordering products can approach sales forecasting. They do not have to forecast sales of a particular product (in a specific product type, color and size) right away. They can first prepare forecasts for a category of products, which is a more manageable task, and then divide the category forecast into forecasts for individual products using proposed by us approaches. This removes the problem of entering unmeasurable product characteristics, which can have very many values, into the training data. Features such as color, size or product type may not always be part of the constellation of attributes passed to the algorithm responsible for the forecast. This is especially important in the fashion industry, where we encounter frequent product substitution in the offerings and where there is a relatively short and specific product life cycle.

5 Incorporating additional time series related to demand forecasting

Short-term demand forecasting with history data using artificial intelligence methods is an known task. It was many times discussed in a literature that concerned a specific area of interest, e.g. water demand [121], electricity demand [132, 123], food sale [156], retail sale [141, 147], transportation demand [18, 17], natural gas consumption [109] and other. In this chapter only some range of the short-term forecasting for product with history will be addressed. In this chapter, attention will be paid to the issue of influence and how to use data from similar time series. We will be considered the case of adding information from time series from other locations and from the same location for which the main forecast is made.

In many real-life problems and in business scenarios we deal with many similar time-series. For example, we have multiple shops where we sale the same product or we have multiple locations for which we want to predict demand for water, energy or other type of media. External factors may have similar effect on all of the locations (e.g. national holiday may influence bigger sale of the specific products in all locations) but it can also be observe that the effect is different only in some location but in in other location it is negligible (e.g. outside temperature may influence energy demand in household that use electric heating but it may be irrelevant in industrial plants that use energy to operate round-the-clock machinery). With this in mind we may want to identify similar time-series that could be used in order to enrich dataset that will be used to create forecasting model. The natural choice in this case would be using clustering methods in order to group time-series.

For forecasting purposes, clustering techniques are frequently employed. A k-means clustering algorithm, decision trees, artificial neural networks, and support vector machine techniques were all used in the paper [91]. For forecasting electricity load, the authors of the paper [119] described using k-means clustering and an ARIMA model. A hybrid sales forecasting system built on decision trees and clustering was presented in [151]. The paper [105], the author proposed time-series clustering based on common principal component analysis that was inspired by K-Means clustering method. In [7], clustering of time series data was extensively discussed. In this study, there were five categories used to divide the clustering algorithms: Partitioning, Hierarchical, Grid-based, Model-based, and Density-based. The authors of [116] used clustering method on load time series in order to find group of residential electricity customers. Predicting based on similarities was also mentioned in [143]. The authors applied the fuzzy network of comparators (NoC) for the problem of ovulation date prediction. In the paper [88] a benchmark study on time series clustering was performed using UCR archive datasets. The study points out that tested clustering methods vary significantly in accuracy depending on datasets.

Another problem is using multiple time-series from one location when predicting value for the interesting aspect. Because these multiple time-series mostly are dependent from each other, we deal with multivariate time series analyses. It can help to uncover complex relationships and interactions between different variables. Time-series forecasting for multivariate time series was presented in [165], where different neural networks where tested. LSTM method and its variations were used for multivariate time series forecasting in papers [108, 173, 106, 178]. The authors of [171] proposed using graph neural networks. In the article [160], Prophet, VAR (Vector Auto-Regression) and LSTM were tested in order to forecast the severity of the drought over time using multiple time series. In this study the LSTM model worked the best.

5.1 Incorporating data from similar locations

In the first part of research connecting with incorporating data from related and similar time-series, the data from analogous locations where investigated. In this section, the results described in the conference paper [71] in which I am a first author, will be presented.

In these research we wanted to test different possibilities and impact of data clustering in the process of demand forecasting. We looked at various methods for including data from related datasets in the forecasting process and we grouped the measures in multiple ways. Nine different versions of forecasting where tested, where XGBoost algorithm and typical time series forecasting methods (ARIMA, moving average) were used. For experiments, where XGBoost was used, each time series was changed to tabular dataset. The focus was put on clustering and grouping similar datasets and locations, assuming that some of them should be similar in case of demand characteristic. For example, in some datasets demand can be strongly correlated with their previous values and in other datasets, we won't see a similarly strong connection with historical data. We assumed that adding information from similar location about demand can enrich training data and thus improve the results of the forecast. Now, all of nine version of experiments will be presented.

We carried out following experiments:

1. Univariate forecast – experiments "*moving average*", "*arima predictions*", "*Prophet*" and "*separate groups predictions*". In these experiments we tested creating forecasting model based only on historical data from location under consideration. No additional time-series where used. Results from these experiments were our benchmarks.
2. Forecast after putting all datasets into one group – experiments "*one group – predict normalized sale*" and "*one group – add column with prediction*". In these experiments, data from all locations were used in forecasting model. All locations had to describe similar problem (e.g. all of theme where petrol stations), but they were not grouped in any more detailed way.
3. Forecast based on similar records – experiment "*regression knn*".
4. Forecast based on clustering time series – experiment "*partitional clustering – predict normalized sale*" and "*partitional clustering – add column with prediction*".
5. Forecast based on double clustering – experiment "*double clustering*".

All of the experiments will be described in detail.

Method "*moving average*" (*MovAvg*) is a simple forecasting method based on moving average. The window for these calculations depended on the requirements of the problem and was adapted to the dataset under study.

Experiment "*arima predictions*" (*ARIMA*) was based on ARIMA method. In this experiment, `auto.arima` function from the `forecast` R package was used. In this case a size of a training set was also adapted to the problem under consideration.

Experiment "*Prophet*" was based on Prophet method. In the research presented in [110], Prophet outperformed other tested methods in case of ability to handle the

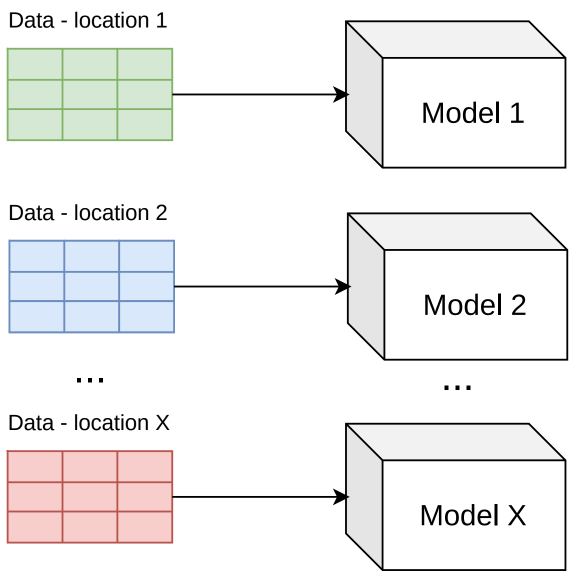


Figure 5.1: Schema of experiment "*separate groups predictions*". In this experiment only historical data from a location under consideration was used in order to create forecasting model for the specific location.

influence of seasonal patterns and holiday effects, so it was hoped that also in this case it will be useful.

In all of the three experiments – "*moving average*", "*arima predictions*" and "*Prophet*" – only the predicted value was taken into consideration.

In the experiment "*separate groups predictions*" (*Sep.Preds*) each dataset was considered independently. A separate XGBoost model was trained for each dataset. A model was trained only on historical data from the specific time series. The schema of the experiment was presented in Figure 5.1.

Experiment "*one group – predict normalized sale*" (*1G.PredNormSale*) is the experiment where each dataset in a first step was normalised. It means that values of a decision attribute were normalised and all attributes created based on a sales history were generated again using normalised values of sale. Then all records from all datasets were connected into one dataset. The model was trained using XGBoost method and then recalculated every specified period in order to detect a possible change in trend. The model forecasted the normalised value of the sales. For each recalculation predictions were made. Each prediction was then reversed from a normalised value to the regular one. The schema for experiments "*one group – predict normalized sale*" was presented in Figure 5.2.

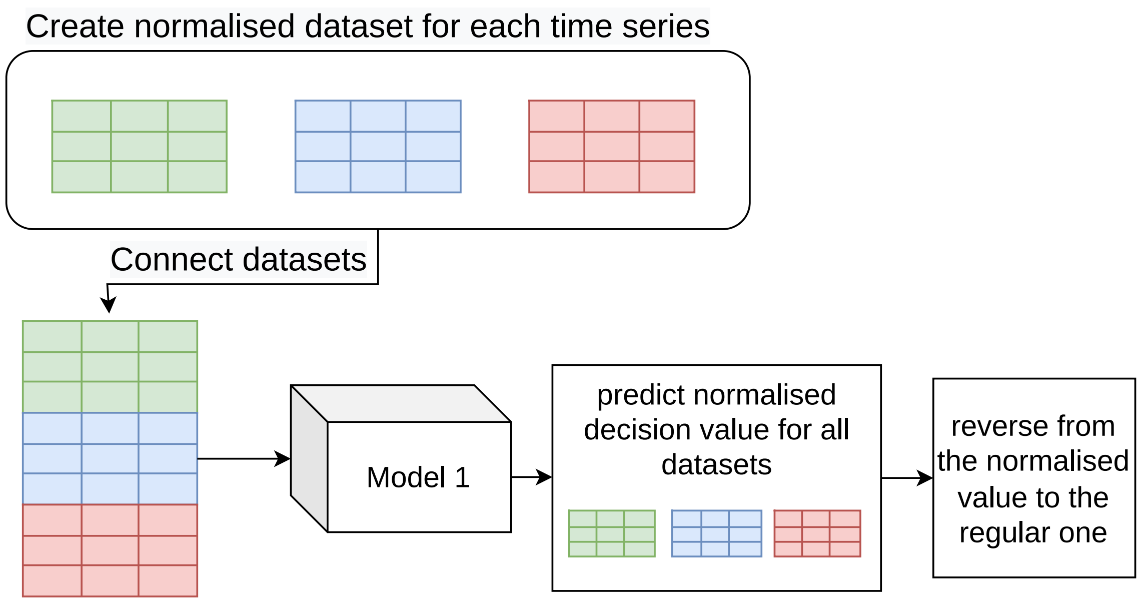

Create normalised dataset for each time series
Connect datasets
Model 1
predict normalised decision value for all datasets
reverse from the normalised value to the regular one

Figure 5.2: Schema of experiment *1G.PredNormSale* where historical data from all locations is used.

Experiment “*one group – add column with prediction*” (*1G.AddPred*) is the experiment where, as before, data from all datasets were normalised and combined into one training set. A model that predicted the normalised decision value was created. Then, for each dataset describing one location (to the train set and test set), a column was added with the forecast from the model (forecast of the normalised value). A separate XGBoost model was then created for each location (the decision column was the actual decision value, without normalisation). The model was also recalculated. The schema for experiments “*one group – add column with prediction*” was presented in Figure 5.3.

Experiment “*regression knn*” (*RegKNN*) used a typical k-NN regression method. Firstly, all datasets were normalised and connected into one dataset. Then we found an optimal number of neighbours for this dataset (parameter `k` for k-NN algorithm) – repeated cross-validation was used on records from 2017-01-01 to 2018-12-31 in order to do this. In the next step, we performed k-nearest neighbour regression that was recalculated every fixed period of time depending on the data under study. Then, predicted normalised values were reversed to the regular decision values. k-NN does not work with missing data so in training datasets only rows with no missing values were used. In a test set, missing values were imputed using the last observed value. The schema of the experiment is presented in Figure 5.4.

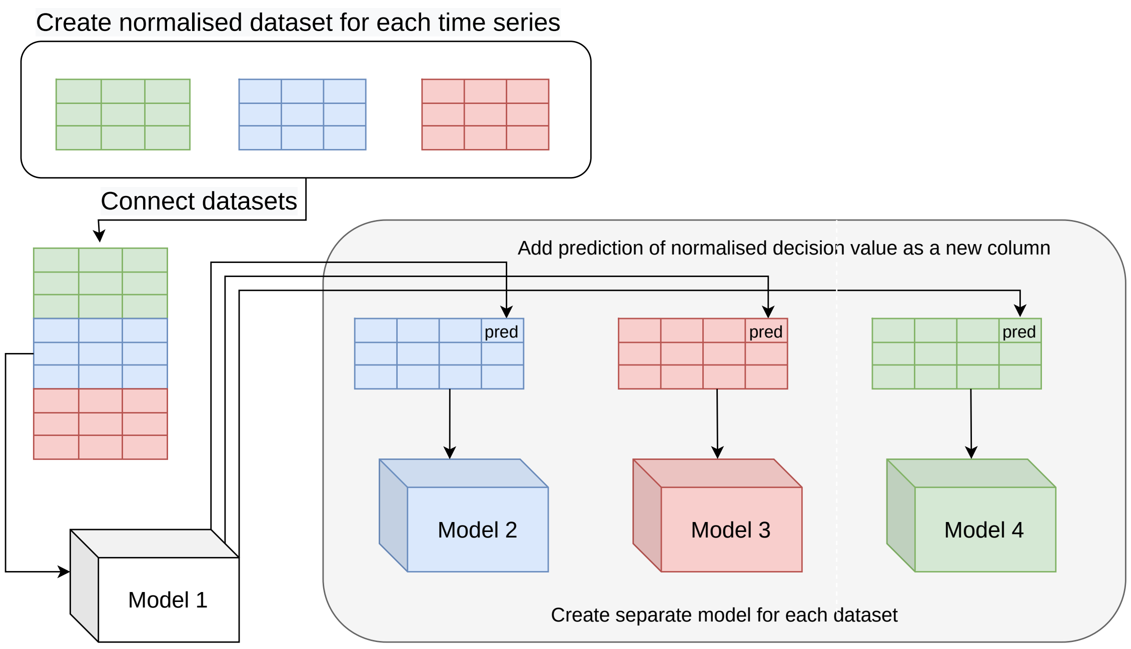

Create normalised dataset for each time series
Connect datasets
Add prediction of normalised decision value as a new column
pred
pred
pred
Model 1
Model 2
Model 3
Model 4
Create separate model for each dataset

Figure 5.3: Schema of experiment *1G.AddPred* where historical data from all locations is used.

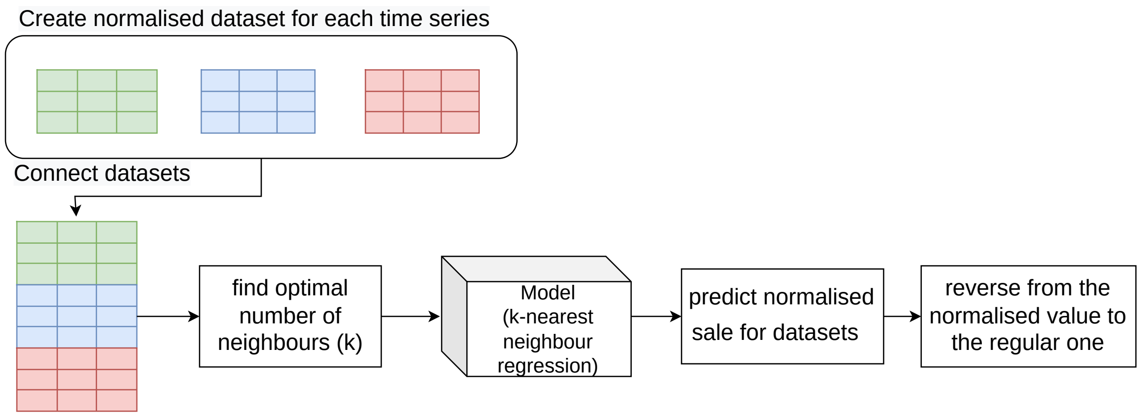

Create normalised dataset for each time series
Connect datasets
find optimal number of neighbours (k)
Model (k-nearest neighbour regression)
predict normalised sale for datasets
reverse from the normalised value to the regular one

Figure 5.4: Schema of experiment *RegKNN* where historical data from all locations are used.

In the experiment "*partitional clustering – predict normalized sale*" (*PC.PredNormSale*) the datasets were grouped by the nature of their normalised time series describing decision value. The `dtwclust` R package and the partitional clustering algorithm were used for clustering. Within each group, all examples from the datasets comprising the group were combined. An XGBoost model that predicted the normalised sales value was created for each group. The final prediction was made based on these models - the output was a normalised forecast, which then had to be inverted to a regular value. For example, assume that we have five datasets describing demand in different locations - datasets *a, b, c, d, e*. In the first step of the experiment, each dataset was normalised and the value of the decision attribute was normalised. Then only time series (not any additional created attributes) were compared and grouped using `dtwclust` package - assume that datasets a and c were classified to group X and datasets *b, d, e* were classified to group Y. Then 2 models were created - the first model that was trained on connected datasets a and c (group X) and a second model that was trained on connected datasets *b, d, e* (group Y). Then predictions were made based on an appropriate model for each data set - predictions from a test set from dataset a were made based on a model trained for group X, predictions from the test set from dataset b was made based on a model trained for group Y etc.

Experiment "*partitional clustering – add column with prediction*" (*PC.AddPred*) – as before, data from all time series were normalised and grouped by the nature of the sales time series. Within each group, a training set was created and a model was trained that predicted the normalised decision value. Then, for each datasets belonging to this group (to the train and test set), a column was added with the forecasts from this model (forecast of normalised decision value). A separate model was then created for each dataset (the decision column was the actual decision value, without normalisation). Showing it on the example: again, suppose that datasets a and c were classified to group X and datasets *b, d, e* were classified to group Y based on their time seires characteristics. Then models were trained for group X and group Y - these models predicted the normalised value of the sales. Then for each non-normalised datasets (*a, b, c, d, e* before normalisation) new column with the prediction of normalised sale was added. So for the non-normalised dataset a, column with predictions from a model created for group X was added. Then for each non-normalised dataset *a, b, c, d, e* separate XGBoost model was trained.

The schema for experiments "*partitional clustering – predict normalized sale*" and "*partitional clustering – add column with prediction*" was presented in Figure 5.5.

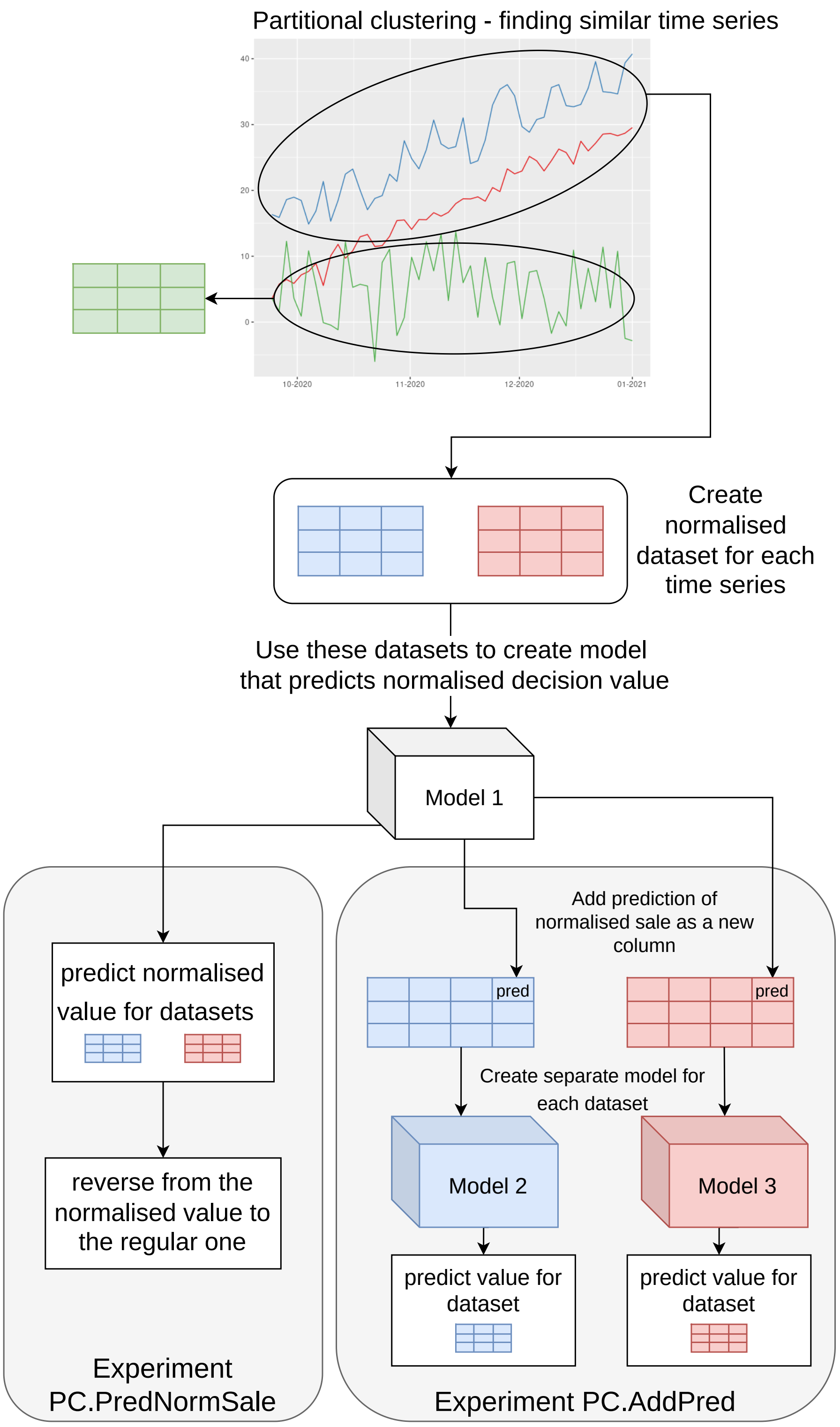

Partitional clustering - finding similar time series
40
30
20
10
0
10-2020
11-2020
12-2020
01-2021
Create normalised dataset for each time series
Use these datasets to create model that predicts normalised decision value
Model 1
predict normalised value for datasets
reverse from the normalised value to the regular one
Experiment PC.PredNormSale
Add prediction of normalised sale as a new column
pred
pred
Create separate model for each dataset
Model 2
Model 3
predict value for dataset
predict value for dataset
Experiment PC.AddPred

Figure 5.5: Schema of experiments *PC.PredNormSale* and *PC.AddPred* using partitional clustering of time series.

In experiment "*double clustering*" (*2Clust*) data from all time series were also normalised and grouped by the characteristic of the time series. Then for each group, a new logical column was added to each normalised dataset. For example, if two groups were created, two new columns would be added to each dataset. The columns indicated whether the row belongs to the group represented by the column. At this point, all records from one dataset would have the same values in these columns – if the dataset was classified to group number 2 then in the second column all records would have a value "1", and in the first column all records would have a value "0". Clustering based on time series and adding columns with information about groups was the first clustering in this experiment. Then all datasets were combined into one dataset. Then an optimal number of clusters for algorithm k-means was calculated and then k-means clustering was performed on the created dataset. This was the second clustering in this experiment. Then for each cluster obtained from k-means, the XGBoost model was created.

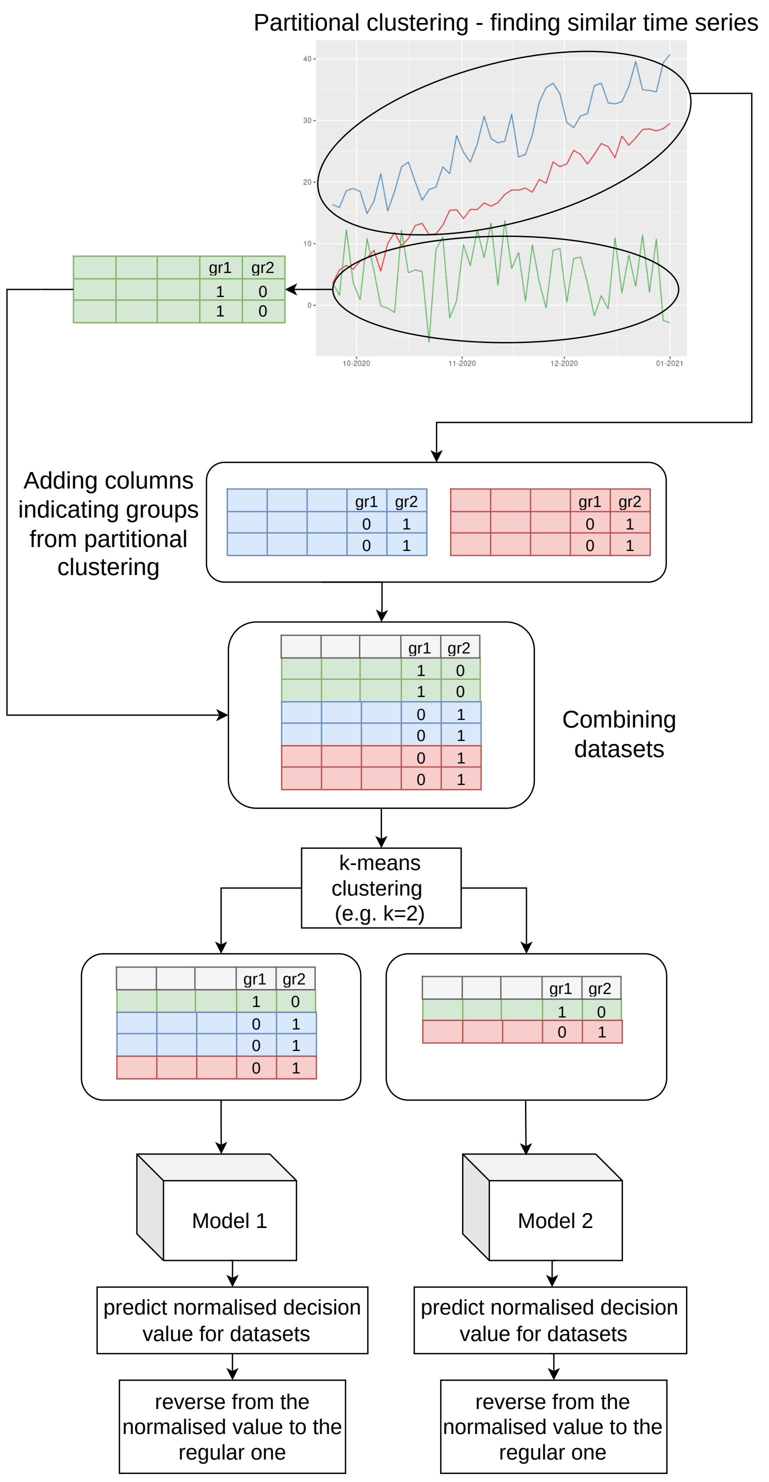
The exceptions were observations that had some missing value. The k-means algorithm does not handle missing values, so some method of data imputation could be used. Here, however, a different approach was proposed – all observations that had a full set of data were grouped by the k-means algorithm and The remaining observations were treated as a separate group. For this separate group the XGBoost model was trained on all the data and it was used when predicting LPG usage for observations with missing data from the test dataset.

It is worth mentioning, that the rows from one dataset could be added to the different clusters from k-means clustering. It means that rows classified into one k-means cluster could have different values in columns representing results from time series clustering. The schema of this experiment is presented in Figure 5.6.

In clustering procedures always normalised decision value and normalised datasets were used.

5.1.1 Datasets used in experiments

The described experiments were performed on datasets from two real-life problems. One group of datsets described sale of fuels on multiple petrol stations. The second group of datasets described usage of liquefied petroleum gas (LPG) in tanks. These two groups of datasets have similar problem – unlike the data that were used in chapters 3 and 4, these datasets did not contain so many additional conditional



Partitional clustering - finding similar time series
40
30
20
10
0
10-2020
11-2020
12-2020
01-2021
gr1 gr2
1 0
1 0
Adding columns indicating groups from partitional clustering
gr1 gr2
0 1
0 1
gr1 gr2
0 1
0 1
gr1 gr2
1 0
1 0
0 1
0 1
0 1
0 1
Combining datasets
k-means clustering (e.g. k=2)
gr1 gr2
1 0
0 1
0 1
0 1
gr1 gr2
1 0
0 1
Model 1
Model 2
predict normalised decision value for datasets
predict normalised decision value for datasets
reverse from the normalised value to the regular one
reverse from the normalised value to the regular one

Figure 5.6: Schema of experiments *2Clust* with two clustering methods

attributes. The datasets contained data descibing decision value, timestamp of the observation and a few conditional attributes. The idea of clustering data from different time series from different locations and using in to train a forecasting model was to enrich simple historical data by additional data. Now the two used groups of datasets will be described and a data preparation process will be presented.

Fuel sales on petrol stations

The datasets were presented in conference paper [71]. We used data from 25 petrol stations located in different part of Poland. All of the petrol stations belonged to one international network of petrol stations. The sale of two most common fuels were taken into account: Gasoline and Diesel. We conducted experiments for 2 fuel kinds at 25 petrol stations, which resulted in 50 tabular datasets that represented the time series of fuels sales. The datasets covered a three-year span from January 2017 to the end of December 2019.

As it was mentioned before, the dataset beside the data describing fuels sale time series, did not contain many additional conditional attributes. In this case only fuel temperature was provided. The raw data derived from stations contained 5 columns:

- *meter* - ID of a fuel tank.
- *start_time* - Start of a considered period of time; timestamp from which fuel sale in a given period is counted.
- *end_time* - End of a considered period of time; timestamp to which fuel sale in a given period is counted.
- *sales* - Total fuel sales (in litres) in a given period of time.
- *temp* - Fuel temperature.

In the first step of the data preparation process, we decided to prepare datasets in a way that they will contain equal periods of time. Most of the periods (*end_time* - *start_time*) last for about 15 minutes, but the periods were not exactly equal. The median time of periods ($n = 5{,}062{,}511$) was equal to 15.27 minutes, the mean value of periods was 16.11 minutes $\pm$ 10.42 minutes. The datasets also contained missing periods, i.e. the *end_time* and the following row *start_time* were not always equal.

Since the experts informed us that predicting for a period of four hours is an acceptable method because it can be used for demand planning, we made the decision

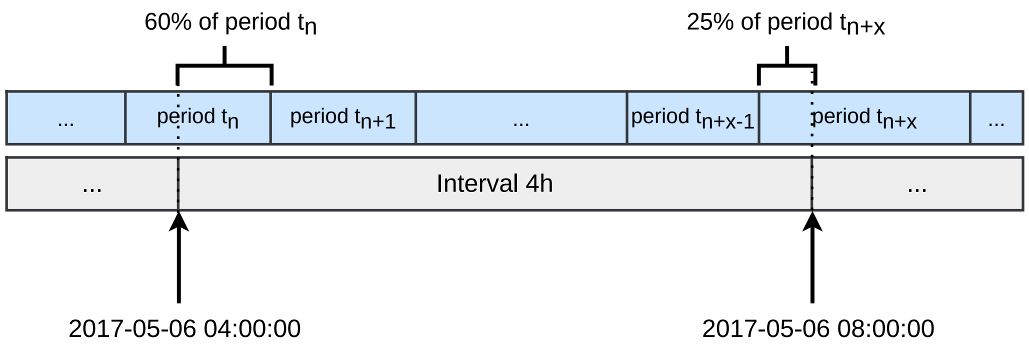


Figure 5.7: Calculate sale value for new intervals in datasets describing fuel sales.

to aggregate sales into four-hour intervals. Forecasting sale for every 15 minutes does not have big practical usage. The sales value for a new interval was calculated as the sum of the sales from all periods that were entirely contained within a given 4-hour interval. To this was added the proportional value of sales from periods that only partially fell within the interval. It is presented in Figure 5.7.

Basic statistics of sales from all of 50 datasets are presented in figure 5.8. They were calculated after creating modified datasets and cleaning the data. The minimum value was not presented because it is equal to 0 in each dataset.

Then, new conditional attributes were created. It was a crucial step because in our experiments we wanted to consider not only typical time series forecasting methods but also method used mostly for tabular datasets. We decided to prepare attributes that could be divided into two categories:

- attributes based on time column and
- attributes based on previous values of sales.

The list of proposed time attributes created based on *end_time* column is as follow:

- basic attributes from time column: year number, quarter number, month number, day number, number of a day in the year, week number, hour, weekday, season, logical attribute describing the weekend;
- logical attributes describing the school holidays: information on whether any school holidays are in progress, information on whether summer holidays are in progress, information on whether winter holidays are in progress, information on whether spring holidays are in progress, information on whether Christmas brake is in progress;

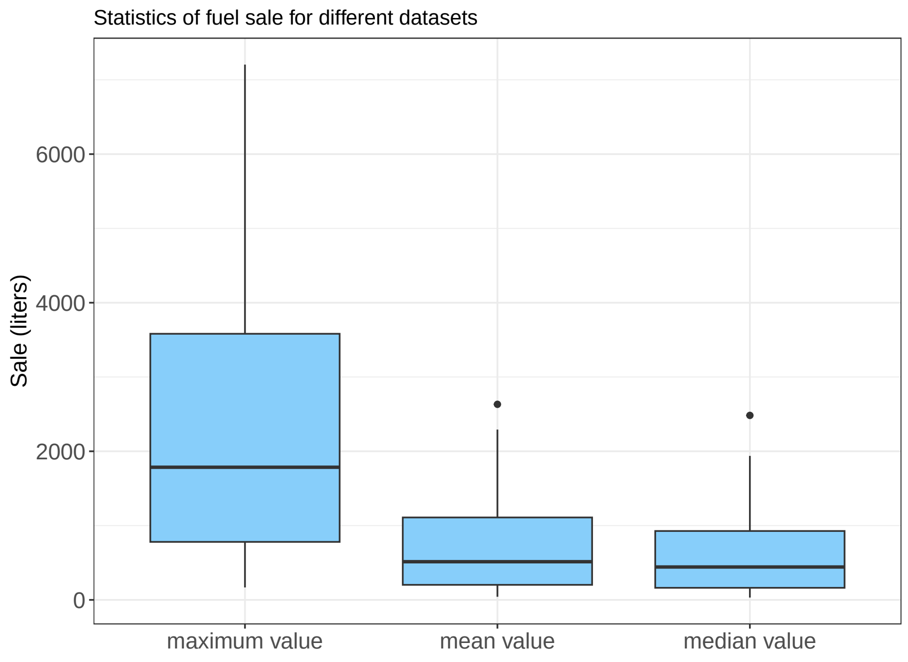


Figure 5.8: A mean, median and maximum value of sales from datasets with 4-hour intervals.

- logical attributes describing the public holidays: information on whether any public holiday is in progress, information about New Year's Day, Easter, Easter Monday, All Saint's Day, All Souls Day, Christmas Eve, Christmas, New Year's Eve;
- cyclical attributes created based on basic attributes from time column - the problem of creating these attributes are described in detail in [24].
- numerical attributes describing a number of days before public holidays (any public holiday, Christmas and Easter) – these attributes have value from a range [0;7].

All considered petrol stations are located in Poland and because of this attributes for public and school holidays were prepared based on polish holidays.

Using forecasting methods typical for tabular datasets has some limitations and problems comparing with time series methods. In a tabular dataset, each row describes one measurement and they are not chronologically connected with each other. In a simple tabular dataset, we lose information about time series. Because of this, we proposed adding attributes describing the history of fuel sales:

- fuel sales observed in previous intervals – from one interval before to six intervals before;
- mean value from x previous intervals (x = (6, 12, 18, 24, 30, 36, 42));
- median value from x previous intervals (x = (6, 12, 18, 24, 30, 36, 42));
- fuel sales observed in previous days in the same hours as a considered measurement - from 2 days before to 7 days before;
- mean value from x previous days in the same hours as a considered measurement (x = (7, 10, 14, 21));
- fuel sales observed x weeks before and in the same hours as a considered measurement (x = (2, 3, 4));
- mean value from x previous weeks and in the same hours as a considered measurement (x = (2, 4, 6, 8));
- attributes describing trend: a preceding interval sales value compared with previous values, a preceding interval sales value compared with mean values from previous intervals, mean values of preceding intervals compared with previous mean values;

The example will be presented in order to better explain creating these attributes. Suppose we want to create attributes for the period where *end_time*= '2019-02-22 08:00:00' (Friday). Some attributes would be as follows:

- fuel sales observed in previous intervals – these attributes would describe sales from intervals '2019-02-22 04:00:00', '2019-02-22 00:00:00', ..., '2019-02-21 08:00:00';
- mean value from 6 previous intervals - this attribute would be a mean value of sales from intervals '2019-02-22 04:00:00', '2019-02-22 00:00:00', ..., '2019-02-21 08:00:00';
- fuel sales observed 5 days before in the same hours as a considered measurement – this attribute would describe sale from interval '2019-02-17 08:00:00';

- mean value from 7 previous days in the same hours as a considered measurement – this attribute would be a mean value of sale from intervals '2019-02-21 08:00:00', '2019-02-20 08:00:00', ... , '2019-02-15 08:00:00';
- fuel sale observed 2 weeks before in the same hours as a considered measurement – this attribute would describe sale from interval '2019-02-08 08:00:00';
- mean value from 2 previous weeks in the same hours as considered measurement – this attribute would be a mean value of sale from intervals '2019-02-15 08:00:00' and '2019-02-08 08:00:00'.

Not all rows from prepared datasetd might be used in the experiments because some of them could have missing values of sales. Also, not all columns were used when training the model. For example columns *id_meter*, *start_time*, *end_time* were not used.

When training a predictive model using XGBoost, the first model was trained on data from 2017-01-01 to 2018-12-31, and then the model was re-trained every 7 days of data. Testing of a given model was done on data that fell between model recalculations.

LPG usage

The datasets describing LPG usage were connected with research descibed in [100]. In the paper, we presented a decision support system (DSS) dedicated to support the LPG supply process by predicting the LPG demand.

We were working on data from 19 tanks. The datasets were obtained from measuring system installed in tanks in which LPG is stored. This system inform what is a gas level in a tank and based on the lowering of the gas level consumption is calculated. Figure 5.9 presents an example plot of LPG level measurements for over a year.

The chart in Figure 5.9 shows a significant and immediate increase in gas level as a result of tank refuelling. Gas consumption is calculated as the difference between the gas level on a given day and the day prior, which yields a highly negative consumption value that ought to be set to zero. Negative consumption but of small value can also result from the change in gas volume due to temperature changes. This phenomena is visible in Fig. 5.9 between months 7 and 10 as very little increase of the gas level. In such case, gas consumption value is also set to zero. The gauge's low sensitivity may also be a problem that can cause incoherent data. This is especially possible for

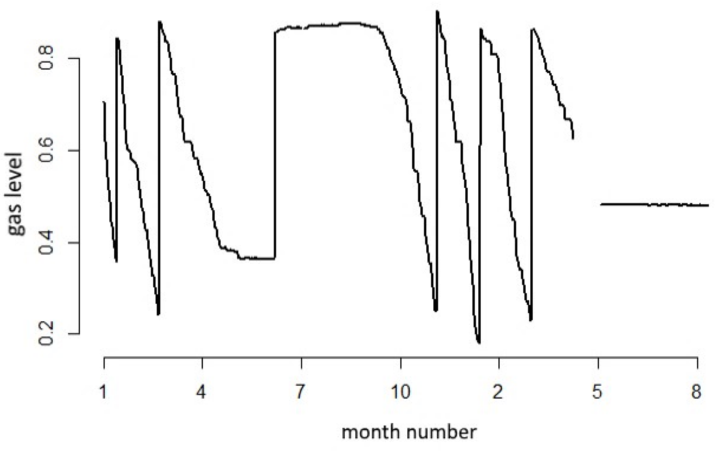


Figure 5.9: Measurements of the LPG level over time are shown as a percentage of the tank's capacity over the course of the year's calendar months.

small households with minimal gas usage. In order to fix this problem averaging the gas consumption values with use of a sliding window was done.

In addition to gas consumption data, which must be pre-processed in an appropriate manner, weather data, specifically weather forecasts, were also used for further research. Weather forecasts had to be used because when forecasting consumption, there is no way to base it on the real temperatures that will be in the future. For this reason, weather forecast data rather than actual weather data should be used to train forecast models. However, because forecasts for the short horizon are usually quite accurate, weather data can be used as an approximation in the training model. Such historical data is also more accessible than historical weather forecasts. The temperature values were used in predictive model because for some locations, especially households that use LPG for heating, the consumption is lower when the temperature outside is higher. It is visible also in Figure 5.9.

After the phase of data pre-processing, we could prepare data for machine learning task. Each tank was a separate dataset. Each row corresponds to one observation for one day. Then new attributes were created. We decided to prepare attributes that could be divided into three categories:

- attributes based on time column,
- attributes based on previous values of LPG consumption and

- attributes based on temperature

The used attributes were as follow:

- *end_time* – end time of an observation (the timestamp).
- *month* – column with a number of the month of the observation.
- *day_month* – a day of the month of the current observation.
- *day_week* – a day of the week of the current observation. 1 corresponds to Monday, 7 corresponds to Sunday.
- *mean_X_days*, where X is 3, 5 or 7 – these columns stores mean usage for 3, 5, 7 previous days.
- *change_cons_3_5* – change in mean usage for three and five days.
- *change_cons_3_7* – change in mean usage for three and seven days.
- *max_temp* – maximum temperature during the day.
- *min_temp* – minimum temperature during the day.
- *mean_temp* – mean temperature during the day.
- *median_temp* – a median of temperatures during the day.
- *range_temp* – range between maximum and minimum temperature during the day.
- *diff_cons_now_X_day*, where X is a number from 1 to 10 – a difference between consumption for the day of observation and consumption one day before, 2 days before, ..., 10 days before.
- *diff_max_temp_now_X_day*, where X is a number from 1 to 10 – a difference between the maximum temperature for the day of the observation and maximum temperature for one day before, 2 days before, ..., 10 days before.
- *diff_min_temp_now_X_day*, where X is a number from 1 to 10 – a difference between the minimum temperature for the day of the observation and minimum temperature for one day before, 2 days before, ..., 10 days before.

- *diff_mean_temp_now_X_day*, where X is a number from 1 to 10 – a difference between mean temperature for the day of observation and mean temperature for one day before, 2 days before, ..., 10 days before.
- *decision_attribute_X*, where X is a number from 1 to 7 – these columns store values of decision attributes for the future 7 days. Therefore, *decision_attribute_1* contains the value for the first day after observation, *decision_attribute_7* - for the 7th day. In the experiments with clustering data, the decision attribute for the next day (*decision_attribute_1*) was used.

The date range of historical data varied for each tank. For the tank with the oldest history, the data is from 2015-03-22 to 2018-08-26. The tank with the latest history had data from 2016-05-28 to 2018-09-24 When training a predictive model using XGBoost, the first model for LPG consumption was trained on data up to 2017-01-01, and then the model was re-trained every 30 days of data. Testing of a given model was done on data that fell between model recalculations.

5.1.2 Results

Firstly, results for fuels sales are discussed. The errors metrics for each experiment were presented in the table 5.1. The table presents the standard error metrics: *Root Mean Square Error (RMSE)*, *Mean Absolute Error (MAE)* and *Weighted Mean Absolute Percentage Error (WMAPE)*. RMSE, MAE and WMAPE were described in chapter 3. The results presented in the table 5.1 show that the best solution among the tested approaches is for experiment *1G.PredNormSale*.

Additionally, mistakes made by each model was related to real-life context. In order to do this. the absolute errors of predictions were compared with the greatest value of sale from a test dataset for each dataset and experiment. The calculation of this error can be presented as follows $\frac{|pred-real|}{maxReal} \cdot 100$, where *pred* is the predicted value, *real* is the actual value and *maxReal* is the maximum value of the actual sales in the test set. Then, for each experiment, the average for each set was calculated and then the average values from all datasets were obtained. The average real-world prediction errors are given in table 5.2. This error allows us to understand the scale of the error we are dealing with and allows us to answer the question of how significant these errors are in a relation to the domain. Here, the *1G.PredNormSale* approach was also the best one.

Table 5.1: Error metrics for the experiments about incorporating additional time series and clustering of data from different time series conducted on fuel sales data.

Experiment	RMSE	MAE	WMAPE
1G.PredNormSale	347.83	207.22	0.33
PC.PredNormSale	348.31	208.07	0.33
2Clust	353.70	212.21	0.33
Sep.Preds	365.10	222.45	0.36
1G.AddPred	369.58	224.47	0.36
PC.AddPred	383.38	228.98	0.36
Prophet	384.50	249.85	0.40
RegKNN	413.61	262.83	0.42
ARIMA	455.42	301.97	0.48
MovAvg	596.98	445.87	0.69

Table 5.2: Relative errors for the experiments for datasets describing fuel sales.

	mean relative error [%]
1G.PredNormSale	6.59
PC.PredNormSale	6.64
2Clust	6.75
Sep.Preds	7.11
1G.AddPred	7.16
PC.AddPred	7.28
Prophet	7.87
RegKNN	8.38
ARIMA	9.74
MovAvg	14.79

For the reason that many approaches (methods) are tested on multiple data sets, the results for Friedman test and Iman Davenport's correction of Friedman's rank sum test were obtained. Then, *post-hoc* analysis – the Nemenyi test – was used. For the reason that Nemenyi test is considered very conservative, then additionally Wilcoxon test with p-value correction for multiple comparisons was carried out. The correction used was Shaffer's (static) procedure, as in [62].

The Friedman rank sum test and Iman Davenport's correction of Friedman's rank sum test both obtained a p-value of less than 0.0001. Because the null hypotesis that all tested methods are the same could be rejected (p-value is less than $\alpha = 0.05$), the *post-hoc* analysis was performed. The results of Nemenyi test are presented by CD diagram in Figure 5.10. This CD diagram was created for significance level of 0.05. It can be observed that the best approach is *1G.PredNormSale*, however Nemenyi test indicates that there is no significant difference between this approach and *PC.PredNormSale* or experiment with double clustering. It is interesting to observe that method using only data from a specific petrol stations was not included in the best group of analysed methods. Another interesting aspect is that method typical for time series (ARIMA, moving average and Prophet) were included in group of approaches with the worst results.

The p-values of *post-hoc* Wilcoxon tests with correction was presented in Figure 5.11. Wilcoxon test is performed for each pair of competing experiments. In each case the p-value of Wilcoxon test is lower than 0.05, so it indicates that each pair of compared methods is significantly different from each other. Considering the given result, it can be concluded that the method *1G.PredNormSale* is the best on tested group of datasets and is significantly different from the other approaches considered.

The results for LPG datasets were analysed in the same way. The Table 5.3 presents the error metrics calculated for each tested method. It can be observed that a similar set of methods is at the forefront of a given comparison as was the case for the results for the prediction of fuel sales (Table 5.1). The relative errors for predictions of LPG usage is presented in Table 5.4. We can see that also here the best method is double clustering approach.

The statistical tests used for comparing many approaches on multiple data sets were also performed on results of LPG usage predictions. The Friedman rank sum test and Iman Davenport's correction of Friedman's rank sum test both obtained a p-value of less than 0.0001. This denotes that there is at least one approach that performs differently than the rest and so the *post-hoc* test can be performed. The

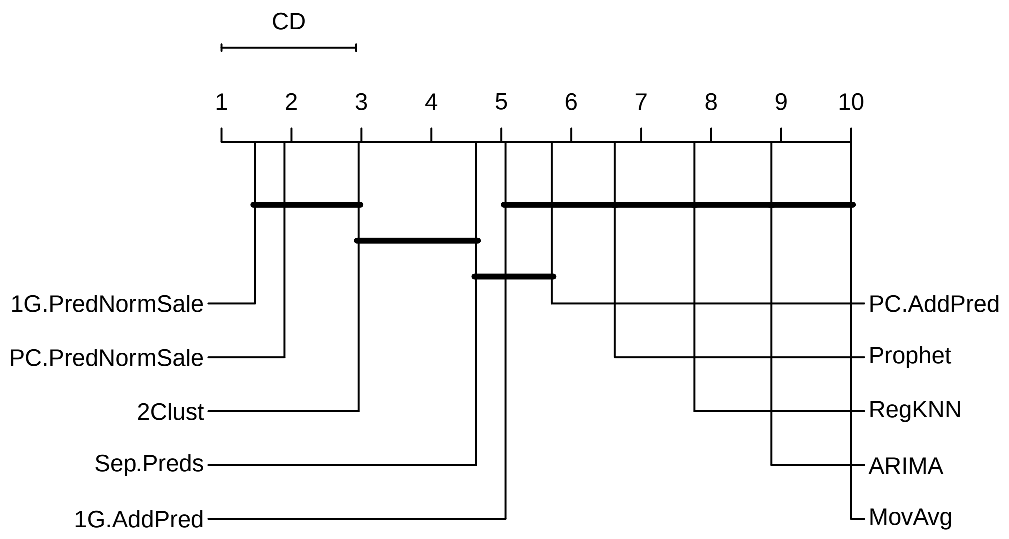
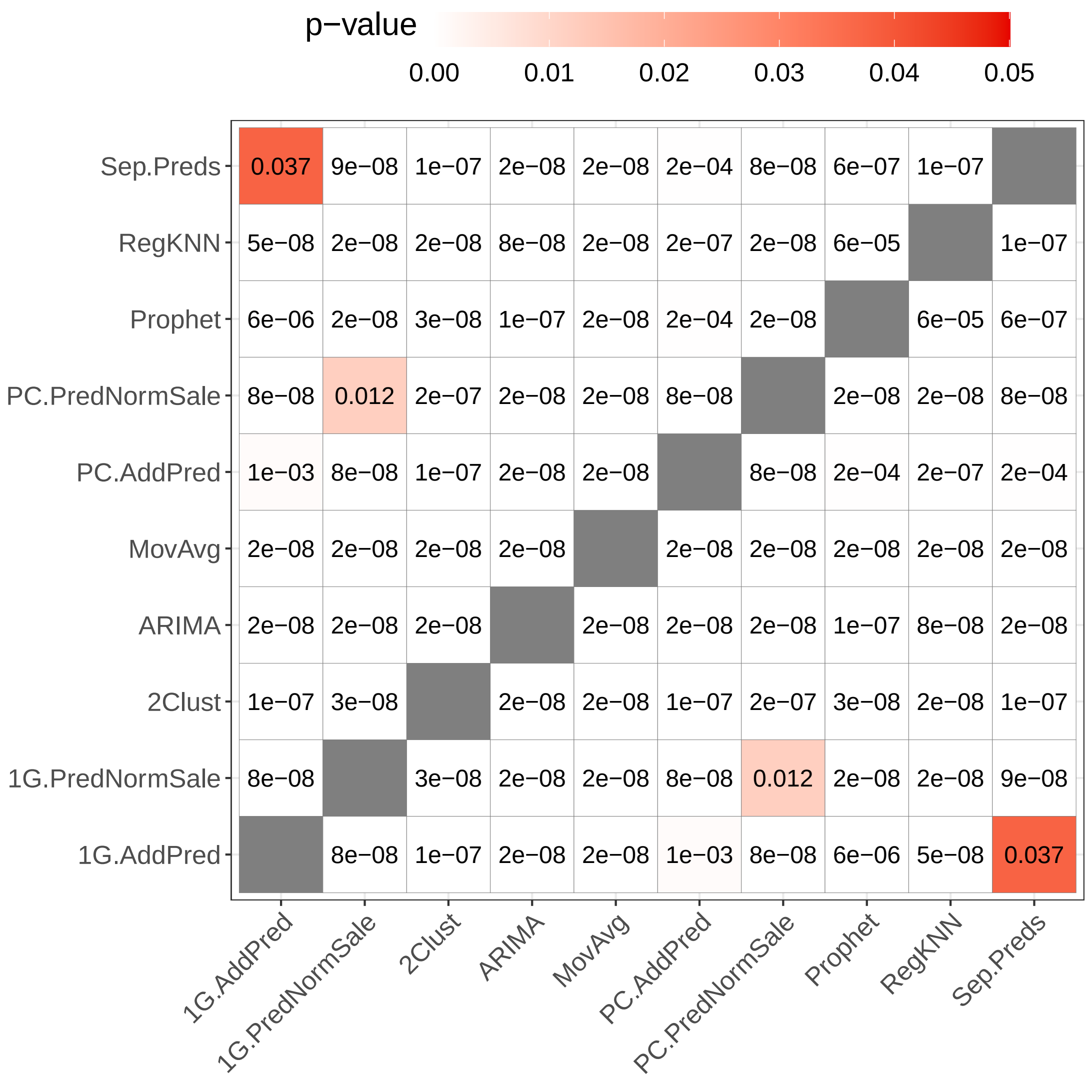


Figure 5.10: CD diagram for comparing proposed approaches for forecasting fuels sales.

Table 5.3: Error metrics for the experiments about incorporating additional time series and clustering of data from different time series conducted on LPG data.

Experiment	RMSE	MAE	WMAPE
2Clust	133.07	52.12	0.57
1G.PredNormSale	133.02	52.27	0.58
PC.PredNormSale	133.21	52.43	0.58
ARIMA	129.42	54.83	0.62
MovAvg	131.60	59.00	0.66
RegKNN	139.94	62.90	0.69
Prophet	138.87	63.27	0.70
Sep.Preds	178.06	63.98	0.77
1G.AddPred	175.78	65.99	0.80
PC.AddPred	178.82	66.54	0.83



	1G.AddPred	1G.PredNormSale	2Clust	ARIMA	MovAvg	PC.AddPred	PC.PredNormSale	Prophet	RegKNN	Sep.Preds
Sep.Preds	0.037	9e−08	1e−07	2e−08	2e−08	2e−04	8e−08	6e−07	1e−07	
RegKNN	5e−08	2e−08	2e−08	8e−08	2e−08	2e−07	2e−08	6e−05		1e−07
Prophet	6e−06	2e−08	3e−08	1e−07	2e−08	2e−04	2e−08		6e−05	6e−07
PC.PredNormSale	8e−08	0.012	2e−07	2e−08	2e−08	8e−08		2e−08	2e−08	8e−08
PC.AddPred	1e−03	8e−08	1e−07	2e−08	2e−08		8e−08	2e−04	2e−07	2e−04
MovAvg	2e−08	2e−08	2e−08	2e−08		2e−08	2e−08	2e−08	2e−08	2e−08
ARIMA	2e−08	2e−08	2e−08		2e−08	2e−08	2e−08	1e−07	8e−08	2e−08
2Clust	1e−07	3e−08		2e−08	2e−08	1e−07	2e−07	3e−08	2e−08	1e−07
1G.PredNormSale	8e−08		3e−08	2e−08	2e−08	8e−08	0.012	2e−08	2e−08	9e−08
1G.AddPred		8e−08	1e−07	2e−08	2e−08	1e−03	8e−08	6e−06	5e−08	0.037

Figure 5.11: P-values of Wilcoxon test with p-value correction for pairs of tested methods for MAE results obtained for fuel sales predictions.

Table 5.4: Relative errors for the experiments for datasets describing LPG usage.

	mean relative error [%]
2Clust	4.12
PC.PredNormSale	4.14
1G.PredNormSale	4.16
ARIMA	4.17
MovAvg	4.55
Sep.Preds	4.88
Prophet	4.88
PC.AddPred	5.01
1G.AddPred	5.05
RegKNN	5.13

results of Nemenyi test are presented on CD diagram in Figure 5.12. The table shows that Nemenyi test also in this case created groups of methods for which no significant differences were found. In this case also for the Nemenyi test it is impossible to point out a unanimous best method. The Wilcoxon test with p-value correction was performed for LPG usage prediction results. The p-value of Wilcoxon tests are presented in Figure 5.13. Unlike the results obtained for the Wilcoxon test for the prediction of fuel sales, in this case test indicated that we could not reject the null hypothesis, so we can't say that tested methods are significantly different from each other. For example, the method *2Clust* has p-value bigger than 0.05 for Wilcoxon test performed on this method and *1G.PredNormSale*, *ARIMA* and *PC.PredNormSale*. The same group of algorithms were combined in one group in Nemenyi test. Given the statistical test results obtained, no single best method can be distinguished for LPG consumption forecasting. However, it is possible to distinguish a subgroup of algorithms with the best prediction results. These are: *2Clust*, *1G.PredNormSale*, *ARIMA* and *PC.PredNormSale*.

Taking into account CD diagrams for comparing proposed methods for predicting fuel sales (Figure 5.10) and LPG usage (Figure 5.12) and analysing the results of Wilcoxon tests (Figures 5.11 and 5.13) it can be concluded that for both cases studied, the best results were always obtained for methods that used data from other locations. In both cases, the best approaches included approaches *2Clust*, *1G.PredNormSale* and *PC.PredNormSale*. For LPG usage predictions also *ARIMA* performed significantly

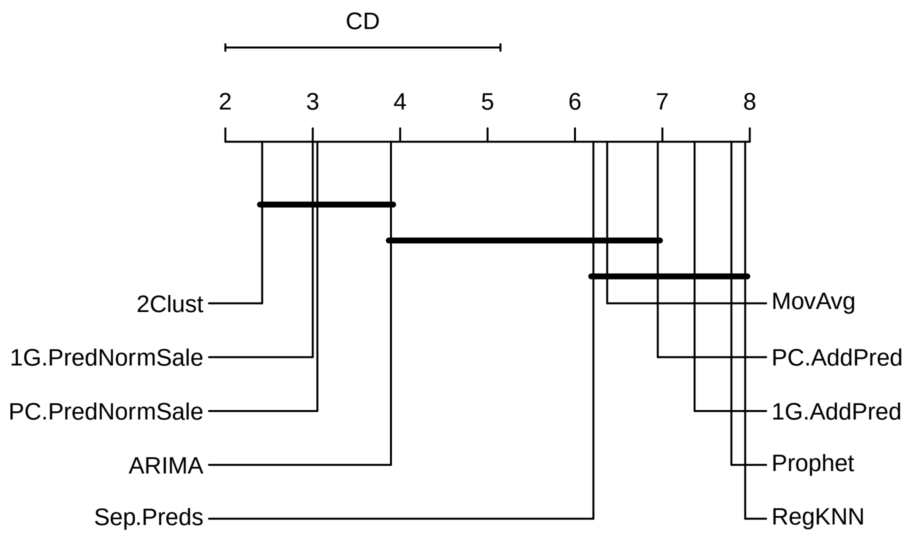


Figure 5.12: CD diagram for comparing proposed approaches for forecasting LPG demand.

well. The best group of algorithms never included a method that used the XGBoost model built only on data from the single location under study.

Considering the given results, it can be concluded that for the cases studied, the introduction of data on a similar problem from other locations significantly improved the prediction results and the given approach may be worth to be applied in similar situations.

5.2 Using data from the same location

Another issue similar to the problem of incorporating data from similar location, is incorporating data from the same location. In this section, the focus will be on adding data from different time series that can be measured in the same location as a predicted time series in order to obtain better forecasts. These issue will be illustrated on the example of forecasting energy consumption in households. The presented research carried out in the context of a project that deals with the development of a digital twin model of a building. Firstly, the problem statement and a research background will be outlined.

Over recent years, the electric energy consumption profile in residential buildings in Poland has experienced more significant changes than in the past several decades

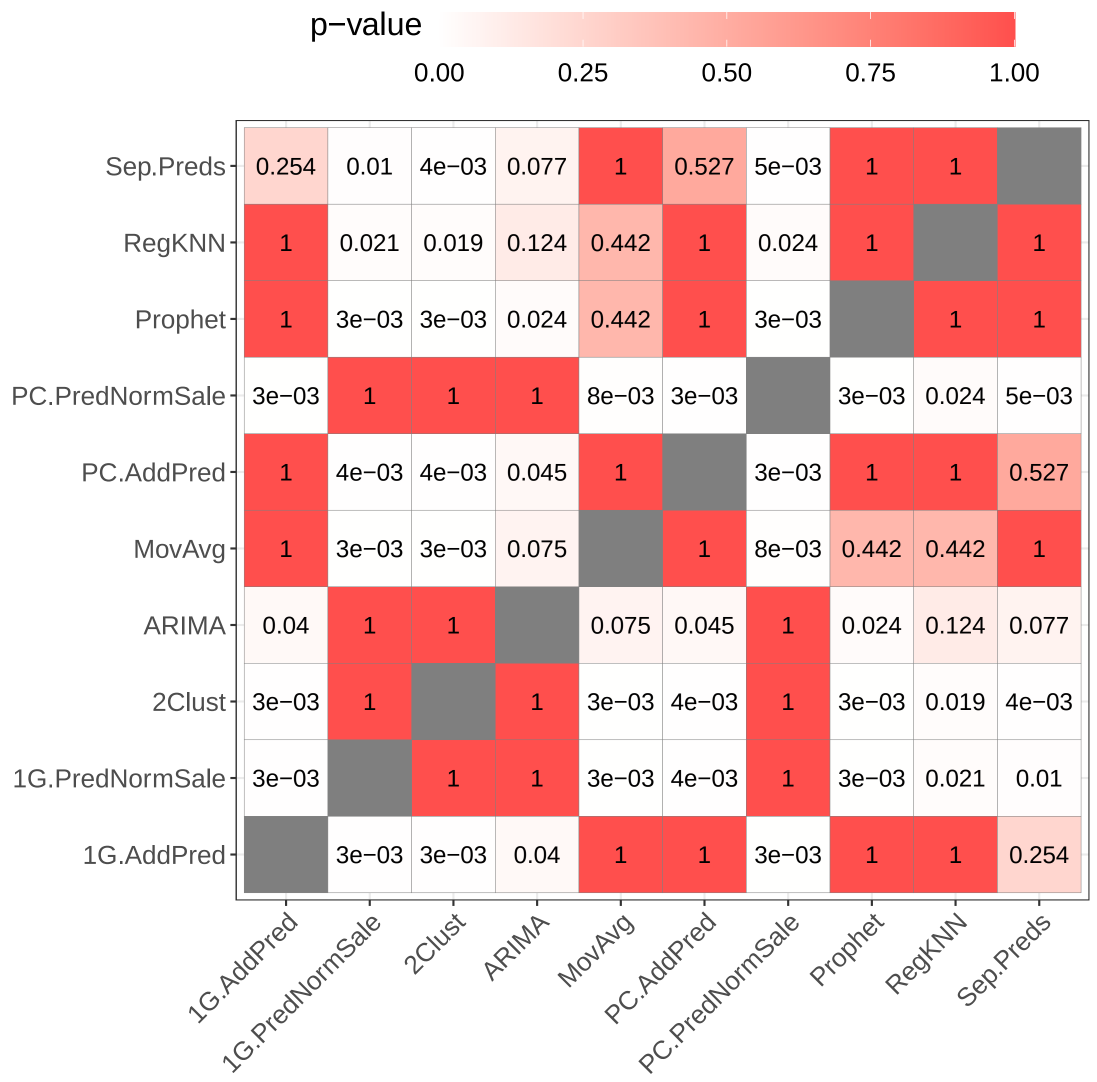


Figure 5.13: P-values of Wilcoxon test with p-value correction for pairs of tested methods for MAE results obtained for LPG demand predictions.

[182]. The most substantial changes have occurred during the epidemic period and the shift to remote work [46]. Factors like the electrification of heating systems (e.g., heat pumps, air conditioning) [86] and social behavior changes (remote work, longer working days) [10] have significantly impacted the daily electricity demand profile.

Electricity demand forecasting is conducted at the power system level [126] by various stakeholders, including transmission system operators (TSOs), DSM/DSR service aggregators, energy sellers and distributors, local government units [124], and large industries [164]. For individual customers in residential buildings, an annual energy consumption forecast is typically sufficient for selecting the appropriate tariff. Because of this, there hasn't been a need for daily electric energy demand forecasting.

However, we can observe the grow of instalations with micro photovoltaic sources in buildings and it is altering the settlement methods for energy fed into the grid from these sources, often making it less advantageous for prosumers (e.g., selling energy at market price). In Poland, the sale of surplus energy from micro photovoltaic sources began in 2022 [1]. As a result, it becomes more cost-effective to use the energy directly in the building or store it in batteries, necessitating daily energy consumption forecasting in buildings.

Forecasting daily electricity demand may also aid in energy consumption reduction strategies and the planning of energy storage utilization. Moreover, it enables the estimation of energy consumption for devices with the highest consumption rates and informs users about their usage. This information is crucial for planning daily energy consumption by individual devices (based on intelligent recommendation systems) in the context of balancing energy with photovoltaics. Optimizing energy usage to be directly consumed in the building, along with solutions that empower smaller prosumers in selling their products [114], can maximize the benefits for prosumers and encourage others to install micro photovoltaic sources.

The research presented in this section are connected with the development of a building model in the digital twin convention [95]. The first part of the project consisted on creating monitoring system of energy consumption in selected buildings. In order to do this, the meters where added to almost all electrical devices. This provided a complete overview of how electricity is used in the building. The schema of the digital-twin model is presented in the figure 5.14. The data about energy usage from different devices were monitored and stored in the databases.

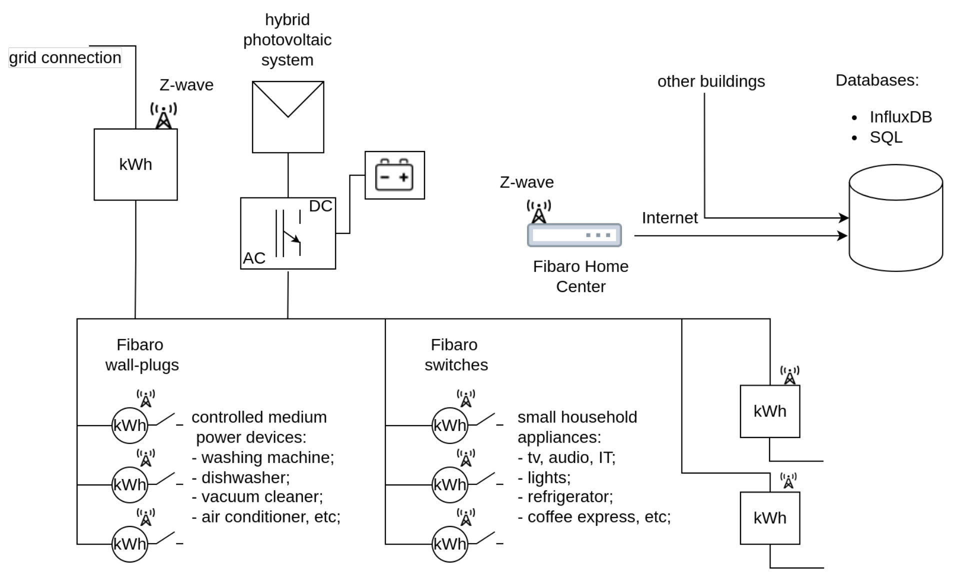


Figure 5.14: Schema of the digital-twin model of a building.

A digital-twin model's ability to continuously analyse data and change daily electricity use in accordance with that analysis is one of its most important features. This feature is especially useful for models that want to reduce electricity use, match solar sources with energy demand, or make use of electricity storage. In these situations, the model can assist in managing how the gadgets in a building operate (by, for example, making recommendations to users). As a result, the module for predicting electric energy demand is crucial to the digital twin paradigm. This is crucial in modern buildings when heating and lighting are provided by electricity, the most cost-effective energy source. By analyzing measured energy consumption data (both current and historical), we conducted prediction studies for electricity demand in a building over the next 24 hours.

The proposed solution was tailored for individual customers, such as households and small service facilities. As a result, it was crucial to create an affordable system, ensuring that installation and operational costs would not exceed potential economic benefits. In this context, costs are mainly associated with the acquisition of measurement and control infrastructure. The proposed solution operates on a limited amount of data—derived from electricity meters, weather forecasts, and calendars. It is worth noting that measuring energy consumption for nearly every device in a residential building is unnecessary, as approximately 20% of devices account for about 80% of electricity consumption. These typically include high-energy-consuming devices, such as heat pumps, electric cookers, washing machines, and refrigerators, which are common across buildings. Thus, measurements should focus on these energy-intensive devices. Another consideration is the energy consumption of measuring systems. Each measuring point consumes around 0.5-1 W, and for 100 devices, this increases annual energy consumption by approximately 400 kWh.

Here we will focus on the module of the digital twin model which is responsible for the electric energy demand prediction. The goal was to obtain less than 25% error of energy consumption prediction. The proposition of approach for forecasting the energy consumption in residential buildings for the next 24 hours will be presented and the usefulness of monitoring the energy consumption of selected household appliances in order to improve the quality of forecasts will be also discussed. The research was presented also in [75]. The problem of energy consumption forecasting was discussed in literature, which will be now presented.

Energy usage can be represented as a time series. Authors of [130] present a literature survey on using ARIMA models for energy demand forecasting, while ARIMA has been utilized in [54, 166]. More advanced methods, such as regression analysis, decision trees, and neural networks, are explored in [155], demonstrating the potential of decision tree and neural network models as alternatives to stepwise regression. Linear regression and multiple linear regression models have been employed for energy consumption forecasting in [23, 41, 77].

In [30], a Bayesian model is created through expert analysis of various factors, including socio-demographic factors, instead of using machine learning or artificial intelligence models. Other approaches, such as agent-based models in [142] and bottom-up methods in [136], have been proposed to model energy consumption profiles but do not focus on time series forecasting models.

In [19], a review of home energy management systems (HEMS) modeling indicates that modeling energy usage for each device is a significant barrier, leading many studies to simplify the problem and use groups of devices instead. This issue could be addressed by incorporating up-to-date data for each device.

Authors of the paper [13] present a regression and statistical technique for predicting energy usage across different building types, focusing on weather conditions as explanatory variables without considering individual device energy consumption. Long Short-Term Memory (LSTM) deep learning models are popular for energy demand forecasting, with various LSTM architectures explored in [113, 96, 26].

Hybrid approaches combining LSTM with other neural network models are also common, as demonstrated in [8, 27, 174].

Alternative neural networks, such as Elman neural networks combined with K-medoids algorithms, have been proposed for prosumer forecasting models in [97]. In [34], a deep learning model based on multi-headed attention and convolutional recurrent neural networks is proposed for residential energy consumption forecasting. However, the dataset used may already be outdated due to changes in energy consumption characteristics, especially after the COVID-19 pandemic altered habits like remote work and longer working days.

Reviews of forecasting in energy storage applications [139] and time series forecasting techniques for building energy consumption [50] emphasize the potential of new AI algorithms and the importance of explainable AI. The Prophet algorithm, used for long-term peak load forecasting in power plants [11] and short-term forecasting for renewable energy sources [159, 68], has not yet been applied to residential energy demand forecasting.

Much of the literature on energy demand forecasting focuses on industries or commercial and educational buildings, with only 19% of studies concentrating on residential buildings [12]. Given that residential buildings represent 21% of total energy consumption in the US, there is a need for more research on predicting energy consumption in residential homes.

Considering related works, our research included some unique properties. As it was mentioned in [19], many studies of forecasting energy consumption simplify the problem due to lack of data about energy consumption of devices in the location. In our study we used the idea of digital-twin model and thanks to this we were able to use enhanced information about energy consumption. These data, with information about whole energy consumption of the location and the weather data, were used in forecasting approaches. As far as we know, this approach is unique in comparison with other work. However, having in mind that monitoring each device may cause a significant increase in electricity consumption, we used only data from selected numbers of household equipment. Additionally, we proposed a methodology for explaining the model in the interpretable way. As it was mentioned in [12], a lot of data-driven prediction models are black-box models, so they provide limited understanding of the situations, when the model makes a mistake. Then, the explanatory methodology was used in order to limit the number of monitored devices. In the research, we used different forecasting methods: naive and linear regression, highly used LSTM networks (used in [96], [26]) but also Prophet method [2] that, to the best of our knowledge, was not described in the literature in the context of forecasting energy demand in the buildings before our paper [75].

This section considers the case of forecasting total electricity consumption using additional time series that make up the given value (we use time series of energy consumption of individual household appliances). However, it is worth noting that the given approach is not limited to this issue. For example, the total sales of products in a given category could be predicted using data from a time series describing total sales and data from the time series of sales of a few of the most popular products, without using detailed information about the sales of sporadically purchased products. This example is not ideal, however, because for forecasts used to supply stores, we are more concerned with forecasts of individual products, not the entire category. This was described in the chapter 4. Another example would be a data centre that needs to forecast what the demand for their resources will be. The data centre serves many users, some of whom are regular customers who have many of their own services running on the servers. Other users are less important, i.e., they use resources to a small extent, for short periods of time or frequently suspend their services. In a total resource demand forecast, one could try to use data from a time series of total resource consumption, but also information on resource consumption by all clients or only selected, most relevant clients. Using data only for selected customers and not all of them (of which there may be thousands) will affect the speed of training the prediction model.

5.2.1 Used datasets

In our study, we examined data from seven different locations, equipping devices with energy consumption meters at each site. Energy consumption for the entire location was also monitored. Information about energy usage was saved in a database every few seconds, stored in InfluxDB, and visualized in the Grafana environment.

Due to errors in the data collection meters, the studies focused on five different locations, which we refer to as A, B, C, D, and E. The characteristics of these locations are as follows:

- Location A: An apartment in a block of flats, housing a family of three (2 adults and 1 child).
- Location B: An apartment in a block of flats, occupied by two adults.

- Location C: A modern detached house, approximately 120 m^2 with electric heating, housing a family of three (2 adults and 1 child).
- Location D: A detached house, approximately 140 m^2, housing a family of four (2 adults and 2 children).
- Location E: A commercial property (office).

Four of the locations are residential buildings, and one is a commercial property. All buildings are located in the Silesian voivodeship. The collected datasets cover the period from March 1, 2021, to October 28, 2021. However, due to missing measurements and data errors, particularly from the first few months of data collection, the specific periods of data considered vary for each location.

In [75] only results from location A, B, C, D were presented because the paper focused on forecasting energy demand in households. Here, also result for one commercial property (location E) will be presented.

The digital-twin model was tailored for specific locations, meaning that for locations C and D, it was created for the entire building, while for locations A and B, it was designed only for specific apartments (not the entire building) and in location E – only to specific office. However, to simplify the terminology, we will continue to use the term “digital-twin of the building” regardless of whether it was created for a house or an apartment. Furthermore, energy consumption predictions were carried out separately for each location. For locations C and D, the goal was to predict the energy demand for the entire building, for locations A and B the aim was to predict the energy demand for the individual apartment and for location E we wanted to predict the consumption for the specific office.

The summary of a type of location and characteristic of observed daily energy consumption can be found in table 5.5. As it can be seen, energy consumption is higher for houses than for flats.

Data preparation

Energy consumption data were consolidated into hourly intervals for each location. By using Grafana for data aggregation, energy consumption information was stored in an incremental format. This approach allowed us to impute missing data since we had the value before the data gap and the incremented value after the gap. Based on this information, we were able to determine the amount of energy used during the

Table 5.5: Summary of location characteristics. The type of location, minimum, mean, median and maximum value of observed daily energy consumption is presented for each of them.

Location	Type	Min. energy [kWh]	Mean energy [kWh]	Max. energy [kWh]
A	Flat	2.14	6.64	13.89
B	Flat	1.34	3.66	10.44
C	House	3.92	17.19	29.68
D	House	6.68	14.65	34.62
E	Office	2.69	4.97	8.14

missing period. Missing data were imputed using linear interpolation, and we also documented which hourly consumptions were imputed and which were the actual values from the database.

Next, the historical data about weather were added. This data was used because it was assumed that if a building is heated with electricity or has air conditioning in it, the weather will affect energy consumption. We used the information about the temperature that was measured in each hour and the percentage value of the overcast. The historical weather data were the same for all locations because all of them were located close to each other.

The data had to be aggregated into the daily periods because the goal of our research was to forecast energy consumption for the following day. We computed new attributes detailing the lowest, maximal, and mean temperatures and overcast values for each day, as well as the proportion of imputed values for each day. We then totaled up the energy consumption for each day. The days for which there were less than 24 hourly observations (mostly the first and the last day from the whole dataset) were removed from the datasets.

In the figure 5.2.2 we present the decomposed energy consumption time-series for each location. We present the trend and weekly seasonality components of the time-series. We can observe that the characteristic for each location, especially in terms of weekly seasonality, is different for each location. In the office (location E) is visible a significant drop in energy consumption on weekends, which is in line with intuition.

In our research, we sought to employ LSTM models, which learn from historical data. For each data point, a new vector of historical values (inputs) must be generated.

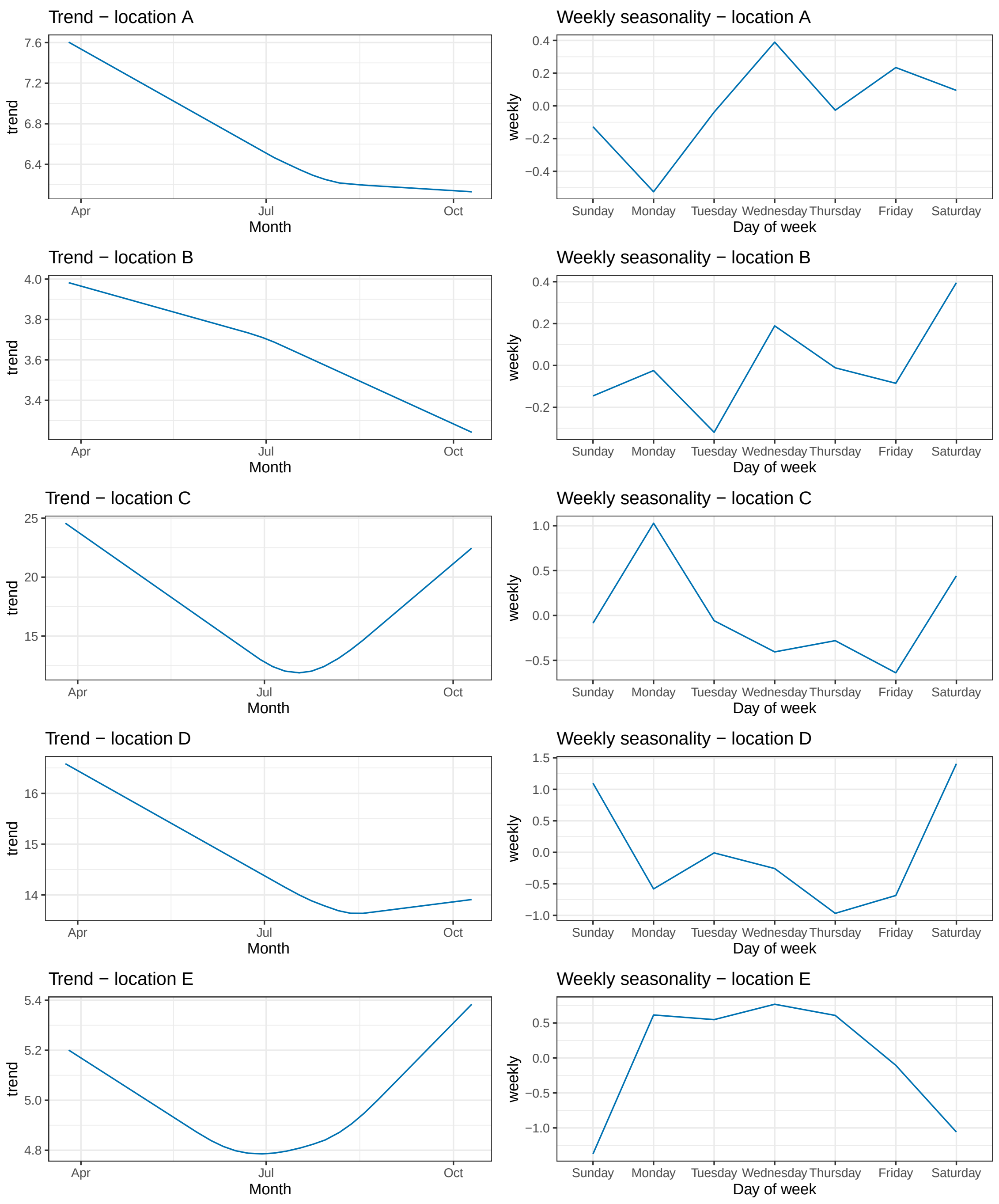


Figure 5.15: Decomposition of energy consumption time-series for each location. Trend and weekly seasonality are presented.

We established that our historical horizon would be equivalent to two weeks, meaning the model would learn based on the previous 14 values. We assumed that predictions should be made for the following day and that users should receive their usage forecasts for the next day no later than 6 p.m. Consequently, when making predictions for a specific day, the actual value from the day before the considered day could not be used for training and recalculating the model. This results in the input for the LSTM model having values ranging from 15 days to 2 days before the considered day. The model's output was the energy consumption for the specific day. For each available day, we created a row containing all inputs (14 values) and output (1 value).

We employed information about the percentage of imputed values to decide which of these rows could be used in our subsequent experiments. All rows for which the output had an imputation percentage greater than 50% were removed from the dataset, as forecasting models should learn from actual observed values in the location rather than imputed values.

Next, we calculated the mean value of the percentage of imputed values for the inputs for each row. If the mean value exceeded 50%, those rows were also removed. We set a 50% threshold because we assumed that if more than 50% of the values used to calculate daily energy usage were imputed, we would not have a realistic value for that day. We aimed to minimize situations where the model learns from an excessive amount of artificially entered data while also avoiding the deletion of too much data.

In the next step of data preparation process, the data of the energy consumption for the top 10 most energy-consuming devices was added to the datasets. These top devices were selected based on their total energy consumption calculated across the entire available dataset. As a result, the energy consumption of some devices, such as air conditioning, varied significantly throughout the months under consideration, while other appliances maintained relatively consistent energy consumption throughout the entire period. Also, for each location a different set of appliances were chhosen. Below, the top 10 devices are presented for each location:

- In location A – computer, shower light, recess lighting, outdoor lighting, dinner room lighting, washing machine, Wi-Fi socket, socket under desk, bedroom lamp, hood.
- Location B – fridge, socket for RTV, dishwasher, socket no. 1, socket no. 2, microwave, socket no. 3, air conditioner, socket no. 4, socket no. 5.

- Location C – socket for hot water tank, heater in the bathroom, radiator heater, fridge, socket for the desk, dishwasher, induction stove, fridge in the pantry, socket under TV, socket in the office.
- Location D – fridge, TV-Audio, dishwasher, fridge no. 2, dryer, boiler, TV in kitchen, kettle, socket near the desk, alarm power supply.
- Location E – rack, refrigerator, dishwasher, outlet near desk number 1, lightning near window, stove, outlet near desk number 2, socket in secretary's office, kettle, main light in secretary's office.

It's important to note that the descriptions refer to the way the database describes each particular socket. However, it is impossible to confirm whether or not the family members have plugged various devices into specific sockets.

Summarizing the used datasets and data preparation process: for each location dataset containing approximately seven months of daily total energy consumption data was prepared. Missing data points were imputed using linear interpolation. Historical weather data for each day, including mean, minimum, and maximum temperature values as well as daily overcast levels, were added to the dataset. Additionally, data describing the energy consumption of the top 10 energy-consuming devices was added to datasets. Furthermore, we retained information on which data points could be utilized for training and testing the created models, basing this selection on the number of imputed values.

5.2.2 Experiments

The aim was to identify the most effective method for making forecast using also additional time series from the same location. As it was mentioned before, here we focused on the problem of predicting the next day's total energy consumption, however we can imagine different situation where similar studies could be performed.

In this study, the target was obtaining mean error of less than 25% when forecasting energy demand for the next day. We sought to determine the optimal algorithm and attribute set to enhance forecasting accuracy. Our experiments investigated how the model's performance varied with the influx of new data, recalculating the models for each new data point when possible. On this data, we assumed that the initial models could be trained with a minimum of 14 data points. Since we decided to use LSTM models with data points where inputs consist of historical data from 15 days prior, we required at least 29 days of data to begin training the model. Models were created independently for each location.

At this point, it is essential to examine why the approach presented in the first part of this chapter – utilizing data from various locations and attempting to group them – was not employed.

The primary reason was the limited number of datasets available; we only had relevant data from five locations. Furthermore, these locations were quite diverse, including two houses, two apartments, and one commercial office, with differing numbers of residents in the residential buildings. Examining Figure reveals that electricity demand varies among locations, such as in terms of weekly seasonal consumption.

Another challenge was that many proposed approaches for grouping data from different locations relied on the XGBoost model, which required tabular data. A single record would need to describe one point in time. To represent all historical data in one record, a set of attributes describing all utilized time series would need to be created, resulting in a significant number of attributes. For instance, LPG data only described the relationships and variations of two time series – LPG consumption and temperature. In the case of energy data, attributes would need to be created for relationships and changes in 11 time series related to energy consumption (1 for overall energy consumption and 10 for individual appliances), along with additional attributes for temperature and overcast time series.

Even if attributes describing changes and characteristics for all time series were created, an issue would remain. Our study intended to use data describing the energy consumption of the top 10 energy-consuming appliances, which differ across locations. If we attempted to create tabular datasets and group data from various locations, the characteristics of one appliance (e.g., a computer from location A) would be compared to the demand characteristics of another top appliance from a different location (e.g., a refrigerator from location B), potentially introducing bias.

Given the small number of datasets used and the numerous time series associated with each location, we opted to concentrate our research on utilizing data collected separately for each location.

The methods used in the approach will be described below.

Baseline and linear regression models

Our baseline model was established using a naive approach. This model predicted the value observed in a given location exactly one week prior. Additionally, we developed four models based on linear regression. The key difference between these models was the quantity of data used to create the linear regression. The data periods utilized were: 30 days, 14 days, 7 days, and 4 days. These periods represent the number of days prior to the forecast day on which the linear regression was determined.

LSTM and Prophet

LSTM and Prophet have been previously presented in earlier sections of this work. In this study, they were employed as more advanced methods for time series forecasting tasks.

In our experiments, for the LSTM network, we used the structure proposed by the Telemanom authors [80]. Our network had 2 LSTM layers: both had 80 units, and after each of them we added the Dropout layer with dropout equal to 0.3. The activation method was *linear*. When compiling, we used *adam* optimizer and mean squared error (MSE) as a loss metric. For fitting the model we used 35 epochs, batch size of 64, and we split the data into train and validation data in proportion 85% to 15%. The training of the model stopped when during 10 epochs the MSE metric was not improving. The LSTM models were implemented using Python and Keras with Tensorflow. For each model created from the deep networks, input values were standardized before training the model. The standardization was based on the training dataset. For the Prophet method, we used the R library version. Prophet allows for multiple time series to be utilized in creating predictions.

To assess the improvement of forecasts with different attribute sets, we considered the following scenarios:

- Using information only about total energy consumption.
- Using information about total energy consumption and weather.
- Using information about total energy consumption and energy consumption of the top 10 energy-consuming devices.
- Using information about total energy consumption, weather, and energy consumption of the top 10 energy-consuming devices.

Additional attributes (weather attributes and consumption of the top 10 devices) were incorporated as new channels in LSTM models and as supplementary regressors in Prophet models. In these instances, we dealt with multichannel time series and multiple input forecasting problems.

As previously mentioned, we sought to evaluate the model quality when new data with potentially distinct characteristics is introduced. Consequently, the LSTM models were recalculated daily whenever possible. The quality of Prophet models over time was assessed using the `cross_validation` method available in the library. Cross validation for Prophet was also conducted daily, if feasible, simulating a scenario where the Prophet model is retrained as new data arrives.

Prophet forecasts also include information about the confidence interval of the prediction. If the confidence interval is too wide (greater than the predicted value), we assume that the model is unable to make an accurate decision (considered as a lack of forecast).

Here, all tested approaches are summarized:

- *val_week_before* – Naive model. It predicted the value that was observed in the location a week before.
- *lr_30day* – Linear regression calculated on 30 days of data.
- *lr_2weeks* – Linear regression calculated on 2 weeks of data.
- *lr_1week* – Linear regression calculated on 1 week of data.
- *lr_4days* – Linear regression calculated on 4 days of data.
- *simple_prophet* – Prophet model that used only information about the total energy consumption.
- *weather_prophet* – Prophet model that used information about the total energy consumption and the weather.
- *devices_prophet* – Prophet model that used information about the total energy consumption and the energy consumption of the top 10 energy-consuming devices.
- *devices_weather_prophet* – Prophet model that used information about the total energy consumption, the weather and the energy consumption of the top 10 energy-consuming devices.

- *simple_telemony* – LSTM model that used only information about the total energy consumption.
- *weather_telemony* – LSTM model that used information about the total energy consumption and the weather.
- *devices_telemony* – LSTM model that used information about the total energy consumption and the energy consumption of the top 10 energy-consuming devices.
- *devices_weather_telemony* – LSTM model that used information about the total energy consumption, the weather and the energy consumption of the top 10 energy-consuming devices.

5.2.3 Results

The results for all 5 locations (A, B, C, D, E) will be presented. The goal of the task was to get the mean error of the predictions lower than 25%. The results of MAPE (*Mean Absolute Percentage Error*) metric for each dataset and each model are presented in the table 5.6.

We also wanted to analyse which method gave the biggest number of days for which the error was below the threshold of 25 %. The results are presented in 5.7.

The conducted experiments and results presented in Table 5.6 reveals that for locations *A*, *C*, *D*, and *E*, at least one approach resulted in a MAPE lower than the 25% threshold. This indicates that the initial goal of achieving errors below 25% has been met for four out of the five locations. For these four locations, the best MAPE results were obtained in the "devices_prophet" experiment, where the Prophet method was employed with information about total energy consumption and the energy consumption of the top 10 energy-consuming devices. What is worth of notice, incorporating additional weather information did not improve the results.

Table 5.6 demonstrates that the best results were achieved for location *E*. This is likely due to the fact that location *E* is a commercial office with consistent energy consumption. This is further corroborated by the fact that an acceptable result of less than 25% MAPE was also achieved for a naive model predicting the same energy consumption as a week earlier.

Acceptable results were not attained for location *B*, where MAPE values for all methods were relatively high.

Table 5.6: The MAPE (*Mean Absolute Percentage Error*) of different models predictions made for 5 locations: A, B, C, D and E. Results that achieve the MAPE < 25% assumption are marked in bold.

Experiment	MAPE A	MAPE B	MAPE C	MAPE D	MAPE E
val_week_before	43.4	47.66	51.58	31.93	**18.77**
lr_30days	33.58	38.93	40.1	29.6	27.24
lr_2weeks	40.86	40.85	41.78	29	33.99
lr_1week	43.91	52.25	35.71	32.32	49.77
lr_4days	47.58	62.01	32.27	35.75	54.36
simple_prophet	34.71	40.69	31.32	27.32	**15.86**
weather_prophet	34.51	40.80	38.93	27.23	**15.53**
devices_prophet	**19.9**	43.90	**18.17**	**11.1**	**6.37**
devices_weather_prophet	**20.81**	44.57	**19.34**	**12.07**	**6.97**
simple_telemony	52.02	49.02	60.54	40.24	**23.83**
weather_telemony	41.54	49.55	51.28	41.13	31.59
devices_telemony	39.69	37.44	70.31	41.53	**24.87**
devices_weather_telemony	56.15	39.14	50.19	37.23	25.26

Table 5.7: The percentage of days for which an error of less than 25% was obtained, determined for each location and each experiment. The results that obtained the highest value for each location are marked in bold.

Experiment	% days A	% days B	% days C	% days D	% days E
val_week_before	47.37	40.4	48.72	58.17	68.7
lr_30days	51.97	41.06	47.86	55.56	61.74
lr_2weeks	50.66	34.44	45.3	56.21	49.57
lr_1week	40.13	29.8	49.57	47.06	34.78
lr_4days	37.09	27.15	48.72	45.75	22.61
simple_prophet	55.73	42.98	60.71	61.44	86.96
weather_prophet	55.56	44	53.33	62.75	83.48
devices_prophet	**71.71**	38.18	**77.98**	**91.5**	**98.26**
devices_weather_prophet	68.21	38.89	76.85	88.24	97.39
simple_telemony	37.5	36.42	40.17	52.29	70.43
weather_telemony	44.08	39.74	35.9	39.22	42.61
devices_telemony	46.71	**49.67**	43.48	54.25	52.17
devices_weather_telemony	41.45	45.03	35.65	50.33	49.57

From Table 5.7, it is evident that the "devices_prophet" models also yield the best results in terms of the percentage of days with errors less than 25% observed. This does not apply to location B.

Based on the results presented in table 5.6 and 5.7 we state that the best approach from the tested ones, for predicting energy usage for the next day is the method that use Prophet model with the information about the total energy consumption and the energy consumption of the top 10 energy-consuming devices. In the Figure 5.16 real daily energy consumption values vs. their predictions are presented. The dashed blue line represents the equation $x = y$. The closer a point lies to this line, the more accurate the forecast value is in relation to the actual value. The charts confirm the results presented earlier – satisfactory prediction results were obtained for locations A, C, D and E. For localization B, it can be seen that the predictions differ significantly from the actual values.

5.2.4 Analyse of made mistakes

As previously stated, an acceptable error rate was set at below 25%. Our goal was to investigate the circumstances under which the best approach – *devices_prophet* – for each location produced errors larger than 25%. We aimed to identify any patterns that would help us understand when models generate accurate or inaccurate predictions. As it was mentioned in paper [12] a lot of data-driven prediction models are black-box models, so they provide limited understanding of the situations, when the model makes a mistake. In our research, we wanted to address this problem.

To achieve this, we devised a classification task with the objective of determining the situations in which the model was able to provide satisfactory predictions (with an error below 25%) and when it was not. For each location and its *devices_prophet* model, we labeled a prediction as *Correct* when the error for the prediction was below 25% and *Incorrect* when the error for the prediction was equal to or greater than 25%, or when the model failed to produce a prediction. These were treated as two labels in our classification task. Next, we created attributes that could describe the energy consumption pattern for a given day. We aimed to analyze if the total consumption for the entire location was unusual, and if the consumption for a specific device from the top 10 energy-consuming devices was atypical. To examine these situations, we generated a decision attribute for the overall energy consumption and

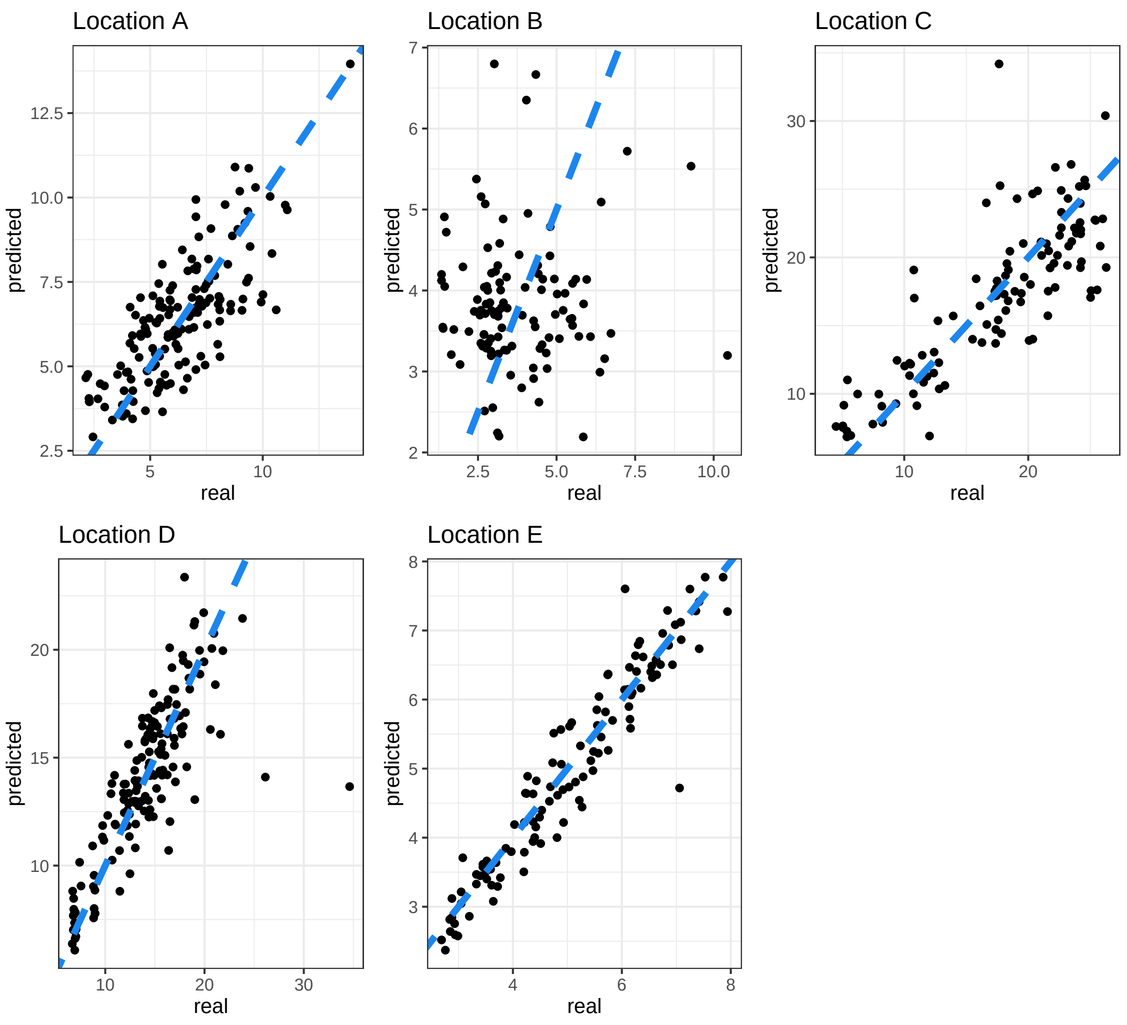


Figure 5.16: The graph shows actual values of daily energy consumption vs. predictions for each location. The dashed blue line represents the equation $x = y$.

for each device that compared energy usage with consumption from different days (from the entire dataset). We identified five different situations:

- Consumption was **minor** – the consumption was in the range $(-\infty, \mu_x - \sigma_x)$, where μ_x is the mean value of the time series x and σ_x is the standard deviation of the time series x. Time series x is the time series describing 14 days before the day for which the attribute value was determined.
- There was a **decrease** in consumption – the consumption was in the range $(\mu_x - \sigma_x, \mu_x - \frac{1}{2}\dot{\sigma}_x]$.
- The consumption was **standard** – the consumption was in the range $(\mu_x - \frac{1}{2}\dot{\sigma}_x, \mu_x + \frac{1}{2}\dot{\sigma}_x]$.
- There was an **increase** in consumption – the consumption was in the range $(\mu_x + \frac{1}{2}\dot{\sigma}_x, \mu_x + \sigma_x]$.
- The consumption was **intense** – the consumption was in the range $(\mu_x + \sigma_x, \infty)$.

Then we created another set of attributes. They described the change of a trend of the time series. We distinguished 5 different situations of trend change:

- The trend was **stable** – the current consumption and the consumption for the day before were described as “standard”.
- The trend was **declining** – the current consumption and the consumption for the day before were described as “minor” or “decrease”.
- The trend was **increasing** – the current consumption and the consumption for the day before were described as “increase” or “intense”.
- There was a **change** in the trend – the current consumption was described differently than the consumption for the day before.

The result was a dataset with 22 conditional attributes – for each of 11 timeseries 2 attributes were created. The decision attribute described whether prophet model made correct or incorrect prediction. These dataset was than used to create a new decision model to determine when the model makes errors.

For the decision model, we used decision trees and their implementation in *rpart* library that is available in R language. We will discuss the results for two locations: B and C. We have chosen them because they represent different effectiveness of *devices_prophet* models. For the location B, the Prophet model gave the worse results among the locations considered (MAPE was equal to 43.90%). For the location C the Prophet model gave relatively good results (MAPE was equal to 18.17%). The generated decision models are presented in figure 5.17 and 5.18.

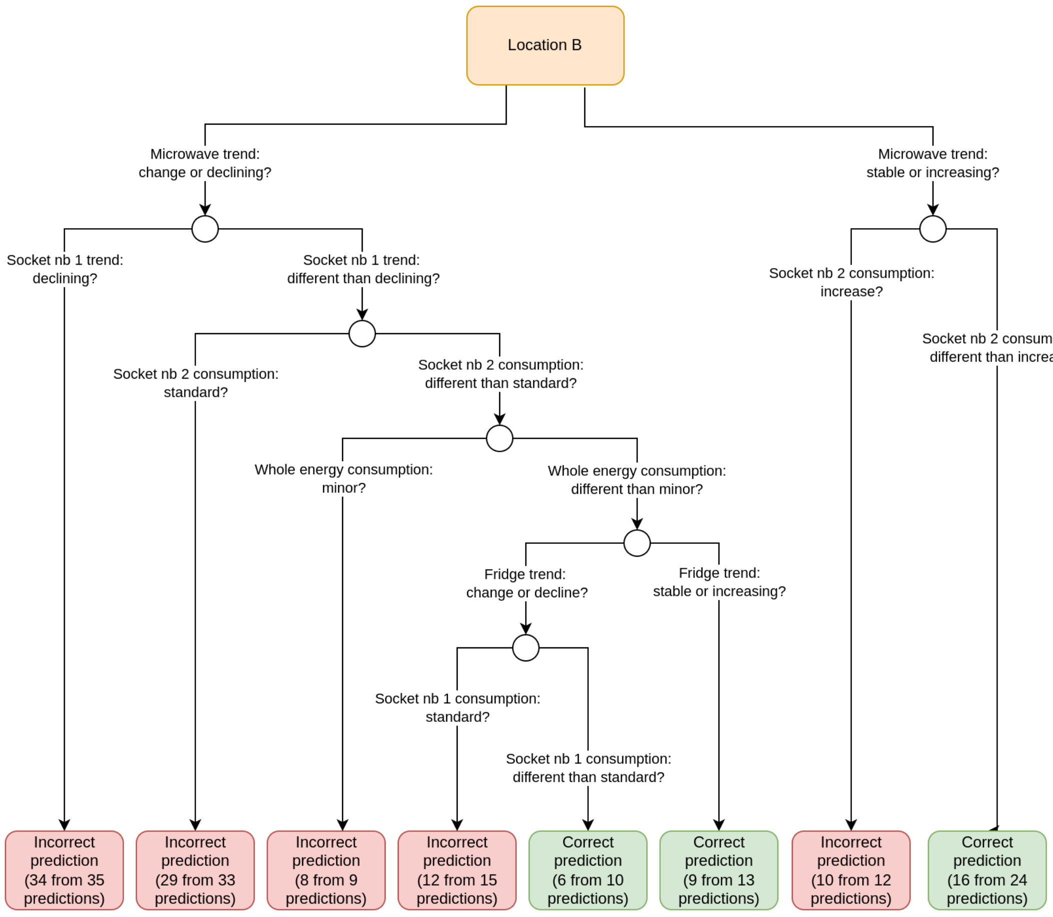

Location B
Microwave trend: change or declining?
Microwave trend: stable or increasing?
Socket nb 1 trend: declining?
Socket nb 1 trend: different than declining?
Socket nb 2 consumption: increase?
Socket nb 2 consumption: different than increase?
Socket nb 2 consumption: standard?
Socket nb 2 consumption: different than standard?
Whole energy consumption: minor?
Whole energy consumption: different than minor?
Fridge trend: change or decline?
Fridge trend: stable or increasing?
Socket nb 1 consumption: standard?
Socket nb 1 consumption: different than standard?
Incorrect prediction (34 from 35 predictions)
Incorrect prediction (29 from 33 predictions)
Incorrect prediction (8 from 9 predictions)
Incorrect prediction (12 from 15 predictions)
Correct prediction (6 from 10 predictions)
Correct prediction (9 from 13 predictions)
Incorrect prediction (10 from 12 predictions)
Correct prediction (16 from 24 predictions)

Figure 5.17: Decision tree explaining when the *device_prophet* models made the mistakes for the location B.

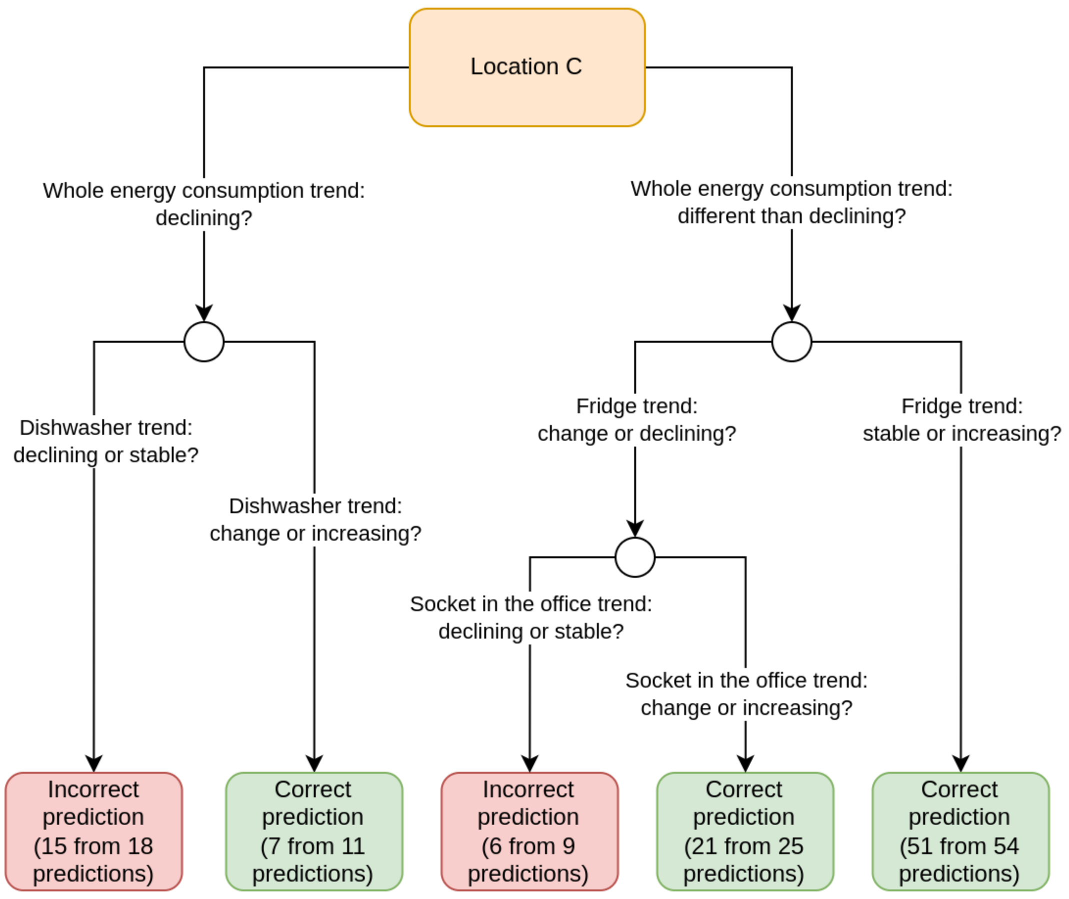

Location C
Whole energy consumption trend: declining?
Whole energy consumption trend: different than declining?
Dishwasher trend: declining or stable?
Dishwasher trend: change or increasing?
Fridge trend: change or declining?
Fridge trend: stable or increasing?
Socket in the office trend: declining or stable?
Socket in the office trend: change or increasing?
Incorrect prediction (15 from 18 predictions)
Correct prediction (7 from 11 predictions)
Incorrect prediction (6 from 9 predictions)
Correct prediction (21 from 25 predictions)
Correct prediction (51 from 54 predictions)

Figure 5.18: Decision tree explaining when the *device_prophet* models made the mistakes for the location C.

The decision tree for the location B has the balanced accuracy equal to 0.80 and the decision tree for the location C has the balanced accuracy equal to 0.79. Balanced accuracy was calculated on the whole dataset. The balanced accuracy of both trees shows that it is possible to find some patterns that describe at which points the Prophet model was wrong and at which points it gave good enough predictions. We can also observe that the decision model interpreting the Prophet predictions is considerably more complex for location B than for location C. The Prophet model for location B yielded unsatisfactory results, which implies that the model could not identify strong patterns in energy consumption at this location. The decision tree model further demonstrates that the model makes mistakes in various situations, and identifying patterns that the model could learn is not a straightforward task.

Even thou some patterns described by decission trees can be difficult for the human to interpret clearly, they can still serve as a way to understand the performance of the proposed predictive model. It can be also used to further investigate the a model or to improve it. As a result, such a model is no longer a typical black-box model.

The above describes a method for detailing the error of an electricity consumption forecasting model, where the main model uses data from multiple time series for a given location. However, it is worth noting that the proposed method of analyzing model errors can also be used in other cases. Firstly, this approach is independent of the predictive model being used. In our case, it was the Prophet model, but it could just as well be another. As long as a percentage error limit can be determined for a given problem, above which the prediction is no longer acceptable, this method of analysis can be used. Based on this limit, a new decision attribute can be created, which will serve to prepare decision trees. Secondly, the proposed method of creating conditional attributes is independent of the data being analyzed. If the primary model used time series from a given location that affect the forecasted value, two conditional attributes can be created based on each such series: a nominal attribute describing the value at a given point in time compared to previous values, and a nominal attribute describing the trend. The same values for nominal attributes that we proposed earlier can be used. As can be seen, when using data from multiple time series from a single location to forecast the main value, the same error analysis can be conducted as proposed above. This is not limited to just an electricity consumption predictive model.

We then decided to use prepared tree models that described the Prophet models errors, to limit number of moniotred devices. We wanted to check whether decision models could help reduce the number of devices monitored at a location.

As mentioned before, each measuring point consumes approximately 0.5-1 W, resulting in an annual energy consumption of about 400 kWh for 100 devices. Additionally, each monitoring device has an associated cost. Reducing the number of monitored devices is beneficial in terms of electricity consumption (borne by the residents of the location) and the cost of purchasing new monitoring equipment (which the installer have to buy).

We hypothesized that some monitored devices, even with high overall energy consumption, may be irrelevant for predictions due to their constant or random energy consumption patterns, which could not be predicted by the Prophet model. We aimed to eliminate these devices from the monitored group and assessed how information about devices obtained from trees describing situations when the Prophet model makes mistakes could be useful for this purpose.

We conducted experiments with datasets for each location (A, B, C, D, E), divided into several train-test datasets. The train-test pairs were divided by the dates: "2021-05-31", "2021-06-30", "2021-07-31", "2021-08-31". This resulted in 20 test cases (4 train-test splits for 5 locations).

For each train-test set, we built a decision tree on the train dataset, describing when the Prophet model made mistakes. Then, on the test part, we examined two cases:

- How will the Prophet model perform on the test part when we use only the time series of appliances whose features appeared in the decision tree and the time series describing the total energy consumption of the location to build a new model?
- How will the Prophet model perform on the test part when we do not use the time series of appliances whose features appeared in the decision tree?

Additionally, we investigated the predictions on the test set when all 10 appliances were used in building the forecasting model.

For each experiment, we built a decision tree for each date in a train dataset and ultimately used the last tree that had a Ballance Accuracy (BACC) ¿= 0.8 and the tree height was greater than 1. That is, for example, if the split between train and test set was for the date 2021-08-31 and the last tree that met the above conditions was built for the set for dates earlier than 2021-08-20, we used the tree built 2021-08-20 and not the one built 2021-08-31.

In summary, for each location, we performed the same process for train-test splits by date 2021-05-31, 2021-06-30, 2021-07-31, and 2021-08-31. The test set was different for each split. The idea is to monitor the total energy consumption for the whole location and the 10 devices during the training period. Afterward, we generate a tree, and the installer either removes monitoring for those devices that appeared in the tree or continues monitoring only those devices that appeared in the tree (two versions of experiments). The third variant is when we do not change anything and continue monitoring the 10 devices used in the Prophet model.

The results of using proposed decision trees to limit the number of monitored devices are shown in Figure 5.19. For each location and split date, the MAPE values are presented. Since the test sets for each train-test split are different, we should compare the three experiment versions for each location and split date rather than comparing the same version across different dates.

In most cases, the version where we removed information about devices that appeared in the tree from the Prophet model yields results very similar to when we monitored all 10 devices. Results for location B can be disregarded since the Prophet model does not perform well initially. In the graphs, we marked our error threshold (25%) with a dashed line. For locations A, C, D, and E, the errors are below this threshold. We assume that such results are due to the fact that the devices that appear in the decision trees have a large variance of energy consumption and therefore are not useful for the Prophet model.

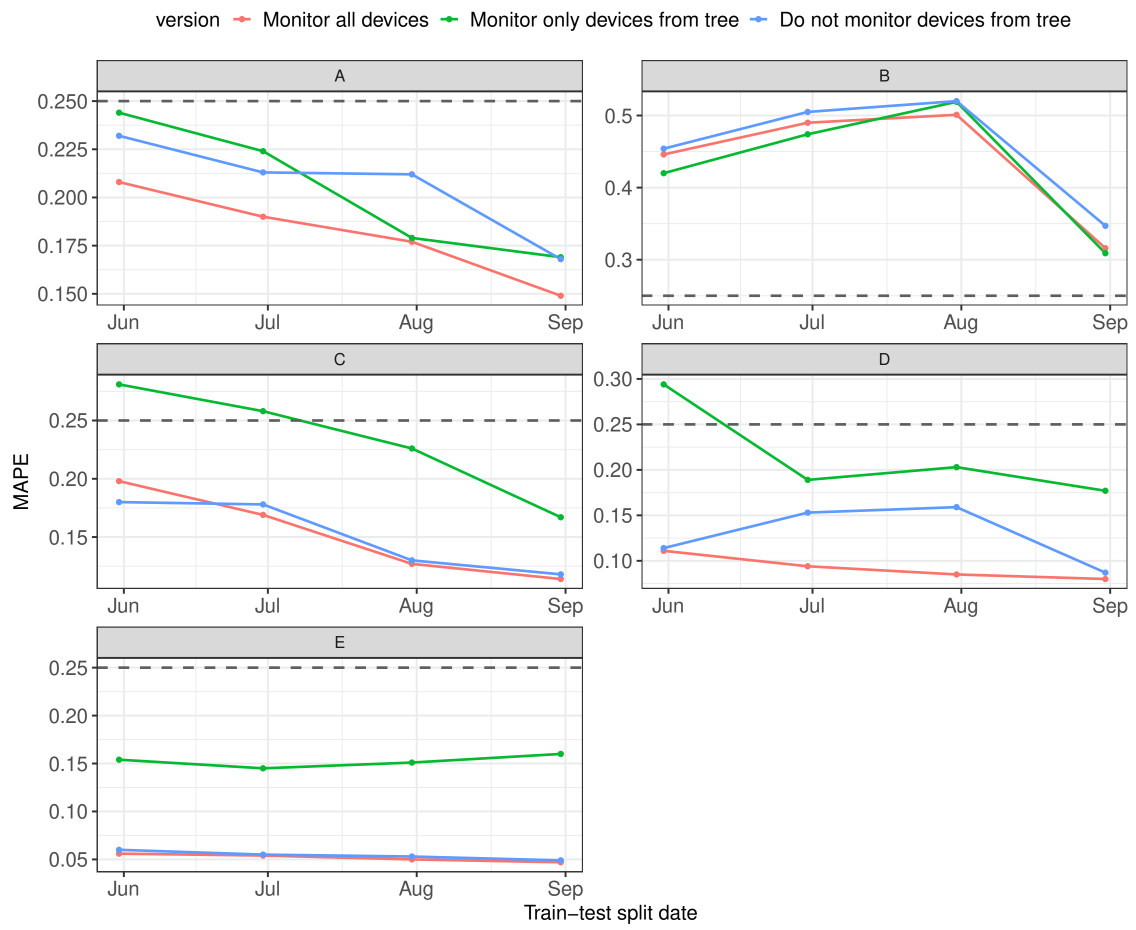


Figure 5.19: Value of an error metric *Mean Absolute Percentage Error* (MAPE) for locations A, B, C, D, E and each split date for train-test datasets. The dashed line indicates the maximum error threshold – 25%.

Table 5.8 shows the number of devices that appeared in the decision trees for each location and split date. In each case, at least one device was included in the tree, and it never happened that all devices appeared in the decision tree. The median of unique devices in the tree is 3.5, the mean is 3.15, and the maximum value is 6. This means we were always able to remove at least one device from the observations, and it never happened that the tree indicated a recommendation to remove monitoring for all devices. The maximum number of devices removed from monitoring was 6. These recommendations seem reasonable and not too extreme.

Table 5.8: The number of unique devices that appeared in the decision trees for each location and each split date.

	Location A	Location B	Location C	Location D	Location E
2021-05-31	4	1	2	3	1
2021-06-30	5	3	3	4	1
2021-07-31	4	3	4	4	1
2021-08-31	4	6	5	4	1

In summary, from a practical standpoint: if an installer wanted to forecast a location's energy consumption, they would begin by monitoring the energy consumption of all N appliances at the location. Based on this data, the Prophet model would be built and start forecasting energy consumption for the following day. With the forecasts, we could assess which days had acceptable forecasts (below the assumed error) and which did not. After gathering information for a longer period, e.g., half a year, the installer could generate a decision tree describing when the Prophet model makes unacceptable mistakes. Devices with characteristics found in the decision tree would be removed from further monitoring, resulting in only M devices being monitored, where $M \leqslant N$. If $M < N$, then residents save money.

5.2.5 Discussion

The potential for enhancing forecast quality by incorporating additional time series information from the same location into the forecasting model was explored.

Experiments were conducted on daily energy consumption data within the context of a digital-twin model of a building. Improved forecasting results were achieved by including information about the top 10 most energy-consuming appliances. According to our research, monitoring energy use across an entire facility as well as for a subset of appliances can significantly increase the accuracy of energy usage forecasts. In the literature we can find lower MAPE results when forecasting daily energy usage, e.g. in [55] MAPE results were 3.11%-5.45%, in [89] obtained results for Artificial Neural Network (ANN) with MAPE 3.5%-9.00% and in [92] obtained MAPE errors were 5.25%-5.43%. However, all of these papers considered the non-residential buildings. For these type of buildings the forecasts may be more accurate due to more predictable energy consumption and the relatively lower variability of occupant behaviour [87]. Also in our studies, for a location E, which was an office, we obtained the best values of MAPE metric. Additionally, for this location even using value from the week before gave satisfactory result with error below 25%. The authors of [85] obtained results with MAPE 12.36% for forecasting the day-ahead energy usage in residential building. It is crucial to note that the results were derived from 3 years of data, whereas our findings were based on less than 6 months of data.

Furthermore, a method for analyzing the forecasting model's errors was proposed. This knowledge can be employed for future model adjustments and for understanding forecasting model. The decision models also reveal which features (utilization of specific devices) have the most significant impact on forecast performance. Last but not least, decision tree models, describing the situations in which forecasting model makes a mistake, were used in order to limit the number of monitored devices.

The proposed research is not limited to electricity forecasting; it can be applied to other cases where the objective is to predict values of a time series composed of values from other related time series. This approach can also incorporate additional time series associated with a specific location. Furthermore, the suggested method for predictive model error analysis is versatile, as the proposed attributes and their nominal values can be defined for any time series.

5.3 Conclusions

The chapter explores the potential of incorporating information from other time series, originating from either the same or different locations. In all examined cases, this additional information led to improved forecasts.

In the first case, we considered the use of time series describing a similar problem from multiple locations (e.g., sales of a product in various stores). We assume that the locations may share similarities due to the nature of demand or sales, but this is not always the case. Therefore, we presented various strategies for grouping data from different locations to achieve better forecasts than when relying solely on data from one location.

In the second approach, we demonstrated the method and advantages of including supplementary time series in the forecasting model, using electricity consumption forecasting as an example. In this case, the additional time series were effectively integrated into the main forecast value. The study utilized data collected during the creation of a digital-twin model for a building, which allowed for the inclusion of energy consumption information for individual appliances. This approach yielded satisfactory results for four out of the five studied locations.

6 Returns

Returns are an important aspect of demand forecasting. It is crucial for proper stocking and ordering new product from a producer. Returned product are mostly resell and underestimating number of returned products may lead to not selling all the products and their consequent backlog in the warehouses. Proper return forecasting can also help to optimize supply chain and logistic processes. The company can thus plan costs for storing returned clothes, refurbishing them and preparing them for re-sale in stationary stores or online. The vendor must also prepare for the additional cost of disposing of returned products, since after they have been shipped and returned several times, the product may not be saleable any more.

The topic of returns has been covered in management and logistics industry literature. The literature review of product returns management was presented in [14]. In an article [134] the authors presents the impact of product reviews on product returns. The product, which has small number of helpful reviews is more likely to be returned because customers buy more substitute products. The authors also analysed the impact of biased ratings on the returns. In the work [131] the authors focus on different sides of returns. They may be due to understandable reasons, for example, the product came damaged, in very poor condition, is poorly made. On the other hand, they can also result from the desire to choose the right size or color of the product, so that the product is purchased in several pieces with similar parameters. In such a case, the possibility of returns is desirable, because this way the customer has decided to buy in the first place and is likely to leave one of the purchased products for himself. Looking at it from this side, it is not surprising that retailers often make it easy for customers to return goods, especially in the e-commerce industry. A store may offer free returns or extend the return period. In paper [120] a systematic review on online retailing was presented. In the aspect of returns, the authors confirm that online retailing leads to higher number of returns than in traditional sale. The effective way of returns management has an impact on customers satisfaction and the probability of their future shopping at the same store.

The review also highlight that the impact of returns managements on customers behavior is different for different product types and it is important to prepare return management according to specific consumer expectations.

In context of fashion retail, returns are also an important aspect. People have different body types, heights and body measurements. Because of this, even in cases where the product has an accurate description on the website and photos that reflect the nature of the product, it can still be returned. There are also customers who, despite buying products stationary, prefer to try on clothes in the comfort of their own home, and for this reason buy many products, with the possibility of returning them later. In article [146] the authors discuss the issue of buying many fashion products online as a result of boredom and impulsive decisions. Returns can allow customers to return ill-considered purchases, but on the other hand, this very mechanism is one of the reasons why a customer may have decided to make such a purchase in the first place. Here, retailers' intentions may vary – they may use facilitated returns to entice customers to buy many more things, but there will also be those who are more concerned with creating a sustainable brand that wants to give their customers the best possible buying experience in their store. Without analyzing the retailers' intentions here, returns are still an important aspect in fashion sales. In [70] authors focus on returns in omnichannel retail. They conclude that adding new possibility of store returns may lead to additional profits for retailers, when customers can have more services without extra cost. In [111] the authors present findings which show that 42.2% of customers, that had an unsuccessful return operation with a seller would not buy again from this same place. Also, both satisfied (80.1%) and dissatisfied (84,4%) clients tend to share their opinion about return experience with other potential customers. From these studies we can draw conclusions about how important the topic of returns is in today's retail sector.

In the work [45], the authors present a data-driven models for predicting return volume at a granular level. The authors used Least Absolute Shrinkage and Selection Operator (LASSO) to obtained the best results. The authors do not focus on predicting if a specific product, from a specific transaction will be returned but they predict a return volume of product type via retailer in specific period. The authors of [65] used neural network for forecasting returns but more for the inventory management of an omnichannel retailer with multi-period sales-dependent returns. The authors highlight that returns is a major challenge for inventory management, due to uncertainty of whether a product is being returned. A literature review indicates that the topic of forecasting returns using machine learning methods, has not been widely analysed and it may be a possible research gap.

6.1 Problem statement

In the research, the possibility of using machine learning for returns forecasting was checked. The aim was to forecast the probability of returning specific product from a specific transaction. Forecasting aggregated number of returned pieces of product in specific period (e.g. during one month) was not investigated. The ability to forecast the return of a specific product can give the knowledge if the product will sell out according to plan. It can also help to maintain good inventory management and help in organizing any needed procedures connected with specific returned product (e.g. cleaning, ozonation). Clients can make returns within a specific period from a purchase, so the probability of return can help to plan all necessary resources in time.

In the research we were working on data from fashion sector. The data were also used in research presented in Chapter 4. The dataset obtained data from the beginnig of 2015 to 10.06.2022. The research was carried out as part of the project co-financed by European Funds entitled 'Decision Support and Knowledge Management System for the Retail Trade Industry (SensAI)' (POIR.01.01.01-00-0871/17-00).

The idea was to create predictive model based on data describing sale transaction. For each product in each transaction we wanted to add decision binary attribute that would indicate if a product was returned. However, after analysing the available data it occurred that in retailer database there are information about sale transactions and return transactions but there wasn't any information which return transaction was connected to which sale transaction. For this reason, there was no way to simply receive information on whether the product was returned. However, we decided to try to link transactions on our own.

A given issue can be interesting from the point of view of many different companies. It is possible that if a given vendor, which has many branches and operates in many countries, does not store a given piece of information, then perhaps many companies may be in a similar situation. A given lack of linkage may be due to historical conditions of the data stored and collected.

6.2 Linking purchase transaction with return transaction

During the process of data analysis, the algorithm of linking sale transaction with return transaction was proposed. The algorithm is presented in Figure 6.1.

In most cases the client has fixed time to make a return after purchasing the product. For this reason, the first step looks for purchase transactions that fall within the stipulated time before the return date. Then the set of possible purchase transactions is limited to only purchase transactions that include all returned products. It is assumed that the return will most often involve products that were purchased at the same time. If, after imposing the given restrictions, only one purchase transaction matched a given return, it can already be considered a matching purchase transaction with high probability. If there were more matching purchases, then further attempts were made to narrow down the set of searches. In the next step, only those transactions were filtered that had the same "location" as the return, i.e. if the return was made online it was assumed that it was also likely that the purchase also took place online. If the return was made in a store, it was assumed that it was likely that the purchase also took place in a stationary store. If this did not result in a single purchase transaction, then it was necessary to return to the previous collection. This is because a purchase made online at a particular company can also be returned in a stationary store and vice versa.

In the next step, another constraints were introduced. The first step checked which purchase transactions had a large number of similar products. Returns in the retail industry often take place because of the purchase of one product in different color variants and different sizes. If many products of the same type were returned, it is possible that even more were purchased. For example, if we have a skirt and pants in a return transaction, it is possible that the skirt in question was purchased in two sizes/colors, and the pants in question were also bought in two sizes or colors. We were looking for purchase transaction with the highest number of similar products. If still not found only one matching transaction it means that there were several transactions that had the same number of similar products as in the return. In this case, the purchase transaction with the highest basket value was selected, and if this also did not lead to a single matching purchase then one was selected at random from the remaining purchase transactions. It is worth noting that the random selection was already done in a heavily narrowed set of possible purchase transactions.

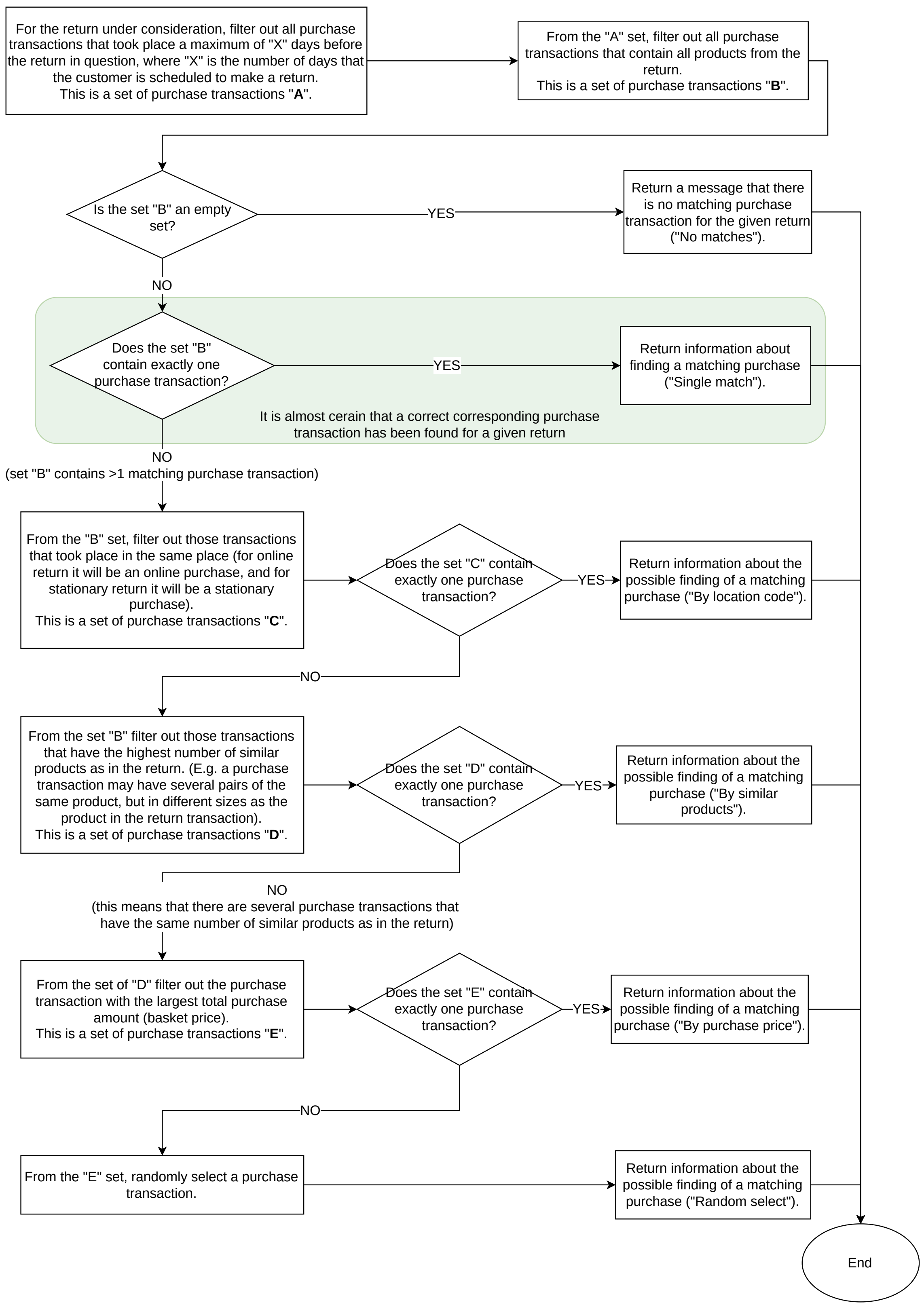

For the return under consideration, filter out all purchase transactions that took place a maximum of "X" days before the return in question, where "X" is the number of days that the customer is scheduled to make a return. This is a set of purchase transactions "**A**".
From the "A" set, filter out all purchase transactions that contain all products from the return. This is a set of purchase transactions "**B**".
Is the set "B" an empty set?
YES
Return a message that there is no matching purchase transaction for the given return ("No matches").
NO
Does the set "B" contain exactly one purchase transaction?
YES
Return information about finding a matching purchase ("Single match").
It is almost cerain that a correct corresponding purchase transaction has been found for a given return
NO (set "B" contains >1 matching purchase transaction)
From the "B" set, filter out those transactions that took place in the same place (for online return it will be an online purchase, and for stationary return it will be a stationary purchase). This is a set of purchase transactions "**C**".
Does the set "C" contain exactly one purchase transaction?
YES
Return information about the possible finding of a matching purchase ("By location code").
NO
From the set "B" filter out those transactions that have the highest number of similar products as in the return. (E.g. a purchase transaction may have several pairs of the same product, but in different sizes as the product in the return transaction). This is a set of purchase transactions "**D**".
Does the set "D" contain exactly one purchase transaction?
YES
Return information about the possible finding of a matching purchase ("By similar products").
NO (this means that there are several purchase transactions that have the same number of similar products as in the return)
From the set of "D" filter out the purchase transaction with the largest total purchase amount (basket price). This is a set of purchase transactions "**E**".
Does the set "E" contain exactly one purchase transaction?
YES
Return information about the possible finding of a matching purchase ("By purchase price").
NO
From the "E" set, randomly select a purchase transaction.
Return information about the possible finding of a matching purchase ("Random select").
End

Figure 6.1: Proposed algorithm of linking return transaction with corresponding purchase transaction.

If at the level of set “B” we have received one matching purchase transaction, then we can say that for this return we are almost sure that we have found a matching purchase transaction. Subsequent steps of the algorithm try to find the most likely matching purchase transaction, but with further steps of the algorithm the probability of a correct match decreases.

Importantly, the given algorithm does not serve to associate with full certainty a return transaction with a purchase transaction and, for example, add the given link in the database as a certain relationship. The given linkage is used to generate a training set that will be used to create a predictive model of returns. For this reason, we accept some bias and inconsistencies with ground truth.

The results of applying the given algorithm will be presented.

We considered 191,999 return transactions.

For 77,936 (40.6%) the algorithm found matching purchase transaction with high certanity. It mans that set “B” from Figure 6.1 had one matching sale transaction so it is very possible that it is a correct matching transaction.

For 5,608 (2.9%) processed returns the algorithm did not find matching sale transaction. It means that set “B” from Figure 6.1 was an empty set.

For 108,455 (56.5%) of return transactions there were several possible purchase transactions. It means that set “B” contained more than one purchase transaction. Further steps made it possible to reduce the given set.

- For 6,851 return transactions there was one matching purchase transaction based on “location”. It means that set “C” had exactly one transaction for these returns.
- For 40,211 returns a match was found by searching for the purchase with the biggest number of similar products as in the return.
- For 59,366 returns a match was found by searching for the purchase with the biggest number of similar products and with the highest purchase price.
- For 2,027 returns a matching sale transaction was chosen by random selection from = a limited set of purchase transactions.

Due to the lack of ground truth about the links between transactions, we are unable to verify the proposed algorithm. However, it can be noted that a large number of return transactions were associated with high confidence with one matching purchase transaction (40.6%). For the remaining transactions, only a small number of them failed to find any purchase transaction (2.9%) and a small number of returns were linked using random selection (1.1%). Taking this into account, the proposed algorithm appears to be a good solution for combining return transactions with purchases when creating a predictive model in cases where there are no clear relationships.

6.3 Creating forecasting model for returns

The aim of forecasting model was predict if the product from a specific purchase transaction will be returned by the client.

One record of data described on product for a specific transaction. Using presented algorithm, the value of a binary decision attribute was indicated for each row. The decisiona attribute informed if the product was returned (value = 1) or not (value = 0) In the first step of data preparation process, the dataset was limited to products that were considered in research presented in 4. Then deleted transactions that contained data gaps or other data irregularities. The result was a dataset that contained 116,789 data records. This dataset was devided into train, validation and test dataset in proportions 70%–15%–15%. A summary of these sets can be seen in the Table 6.1. It is visible that there is a big disproportion between decision classes.

Table 6.1: Summary of train, validation and test datasets used for training and evaluating forecasting model for returns. "Return = 1" shows number of records where decision attribute is 1 (the product was returned). "Return = 0" shows number of records where decision attribute is 0 (the product was not returned).

Set	Return = 1	Return = 0	Number of all records
Train	5,101	76,651	81,752
Validation	1,153	16,366	17,519
Test	1,061	16,457	17,518

In the research three algorithm were taken into account: logistic regression [3], Random Forest [31] and XGBoost.

XGBoost was already described in Section 2.3, so here only logistic regression and Random Forest will be described.

Logistic regression is a classification algorithm for binary classification tasks. It is used to model the probability of a binary response as a function of a set of variables thought to possibility affect the response (called covariates). It means that a probability value is based on one or more independent variables. The logistic regression model calculates the relationship between the independent variables and the likelihood that one of the two possible values will be assigned to the dependent variable. The dependent variable's likelihood of taking one of the two values is represented by the probability score that the model produces, which ranges from 0 to 1. The dependent variable's future can then be predicted using this probability score.

Suppose we have a binary dependent variable Y that takes on the values 0 or 1, and a set of independent variables X_1, X_2, ..., X_n. The goal of logistic regression is to model the probability P(Y=1) as a function of the independent variables. Logistic regression finds an equation of the following form:

$$log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_n X_n$$

where β_0 is the intercept term, and β_1, β_2, ..., β_n are the coefficients or parameters of the independent variables X_1, X_2, ..., X_n. The formula on the right side of the equation predicts the log odds of the response variable taking on a value of 1. The logistic function is then used to convert the log odds to the predicted probability:

$$p(X) = \frac{e^{\beta_0+\beta_1 X_1+\beta_2 X_2+\cdots+\beta_n X_n}}{1+e^{\beta_0+\beta_1 X_1+\beta_2 X_2+\cdots+\beta_n X_n}}$$

Above equation is used to calculate the probability that a given observation takes on a value of 1, and then, using probability threshold, it can be used to class "0" or class "1".

Random Forest is an ensemble algorithm that can be used for classification, regression and other machine learning tasks. This algorithm combines multiple decision trees to make predictions. Firstly, the training dataset is divided into bootstrap samples. These samples are created randomly with replacement, such that each observation has an equal chance of being selected in each sample. each sample is used to build different decision tree. Additionally, at each tree node, a subset of features are randomly selected to generate the best split.. Each decision tree is grown to full depty, without pruning. For new observation, each tree makes a prediction. The final prediction is made by aggregating the predictions from all trees. The algorithm uses multiple trees and random subsets in order to address the shortcomings of conventional decision trees, such as overfitting and sensitivity to small changes in the data.

6.3.1 Approach number 1 – using only original attributes

In the first step, only original attributes describing each product were used. Features have been normalized to a range of 0–1. The conditional attributes that were used are as follow:

- *total_product_price* – the price of the product.
- *category* – the category of the product.
- *size_cdf* – size converted into numerical value. This process was described in 4.
- *color* – color of the product.
- *weekday* – day of the week.
- *sale_month* – month of sale.
- *channel* – sales channel (online or stationary).
- *sale_season* – season of the year.
- *valentines_days* – number of days from/until valentine's day.
- *easter_days* – number of days from/until Easter.
- *christmas_days* – number of days from/until Christmas.
- *womens_days* – number of days from/to Women's day.

6.3.2 Approach number 2 – feature engineering

In the second approach, additional feature engineering was performed. The attributes describing basket and product were added. This is based on domain knowledge, conclusions from the literature review and our own observations of the data – when multiple pieces of very similar products are purchased, or the same ones only in different sizes/colors, the probability of a return on a given transaction increases.

Such a purchase transaction may indicate that already at the time of purchase, the customer assumes that they will leave only one fitting product, and the other similar ones will be returned.

The added attributes were as follow:

- *cart_items* - number of products in the basket.
- *cart_sum* - the value of the basket.
- *basket_same_color_diff_size* - number of products in the basket with the same color but different size.
- *basket_same_size_diff_color* - the number of products in the basket with the same size but different color.
- *basket_num_cat_1* - the number of products in the basket from the category number 1.
- *basket_num_cat_2* - the number of products in the basket from the category number 2.
- *basket_num_cat_3* - the number of products in the basket from the category number 3.
- *basket_num_cat_4* - the number of products in basket from the category number 4.
- *basket_num_cat_5* - the number of products in the basket from the category number 5.
- *pop_score* - the popularity score of the product calculated as the rank of the product sales volume within the category.

In addition to the aforementioned features, features from approach number one were also used.

6.3.3 Approach number 3 – parameter tuning

In this approach, the tuning of parameters for each model was performed using validation dataset. The attributes from approach number 1 and 2 were used. The parameter tuning was performed as full parameter grid search in 5-fold cross-validation mode with AUC maximization criterion. The `GridSearchCV` method from Python `scikit-learn` library.

Logistic regression – tuned parameters

For logistic regression the tuned parameters were:

- *C* – Inverse of regularization strength; must be a positive float. Smaller values specify stronger regularization.
- *penalty* – The norm of penalty. Possible values: *None* (no penalty is added), *l2* (L2 penalty term), *l1* (L1 penalty term), *elasticnet* (both L1 and L2 penalty terms).
- *random_state* – The random number generator used.
- *solver* – Algorithm to use in the optimization problem.

Random Forest – tuned parameters

For Random Forest the tuned parameters were:

- *max_depth* – The maximum depth of the tree.
- *min_samples_leaf* – The minimum number of samples required to be at a leaf node.
- *min_samples_split* – The minimum number of samples required to split an internal node.
- *n_estimators* – The number of trees in the forest.
- *random_state* – The random number generator used.

XGBoost – tuned parameters

For XGBoost the tuned parameters were:

- *subsample* – Subsample ratio of the training instance.
- *reg_lambda* – L2 regularization term on weights.

- *random_state* – Random number seed.
- *n_estimators* – Number of gradient boosted trees.
- *min_child_weight* – Minimum sum of instance weight(hessian) needed in a child.
- *max_depth* – Maximum tree depth for base learners.
- *learning_rate* – Boosting learning rate.
- *gamma* – Minimum loss reduction required to make a further partition on a leaf node of the tree.
- *colsample_bytree* – Subsample ratio of columns when constructing each tree.
- *colsample_bylevel* – Subsample ratio of columns for each level.

6.3.4 Results

The presented experiments were a binary classification tasks. In order to evaluate and compare the obtained results, two method were choosen:

- Balanced Accucary (BAcc)
- AUC

Balanced Accuracy is a metric commonly used in binary classification where imbalanced class distribution occure. In our datasets, there are much more examples where return did not have place than products which were returned. Because of this, Balanced Accucarcy was choosen. This metric can be expressed as:

$$BAcc = (TPR + TNR)/2$$

where TPR is a true positive rate (also known as sensititivy) and TNR is a true negative rate (also known as specificity). TPR can be calculated as the number of true positive predictions divided by the total number of actual positive cases. TNR is defined as the number of true negative predictions divided by the total number of actual negative cases. In our experiments, class 1 was a positive class, because we focused on products for which return has taken place. The higher value of Balanced Accuracy, the better the classification performence. This metric ranges from 0 to 1.

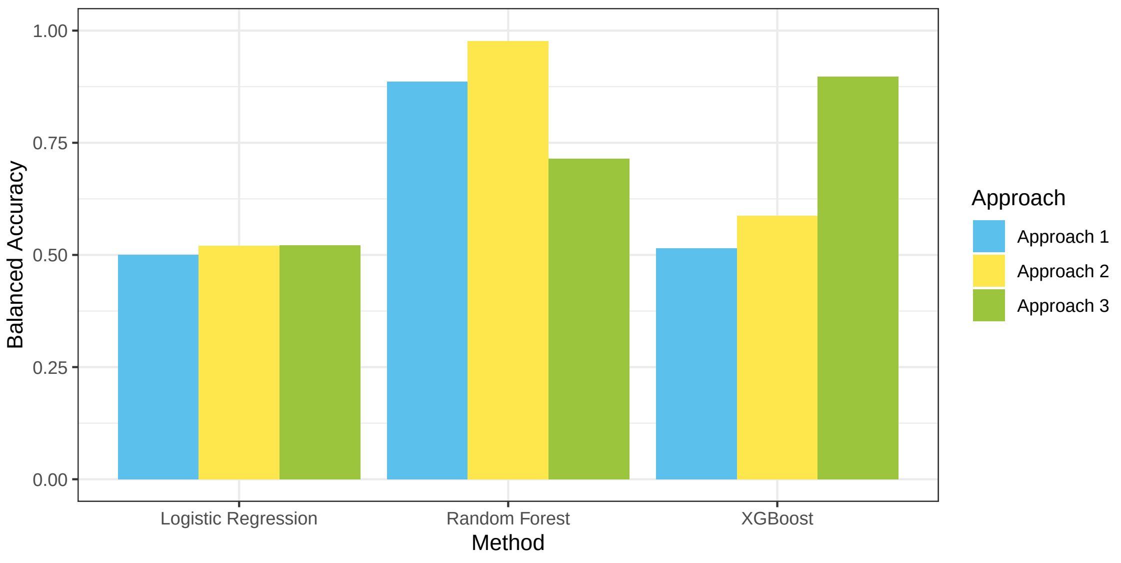


Figure 6.2: Balanced Accuracy calculated on train dataset for each method and each evaluated approach.

AUC (Area Under the Curve) is an another metric used for evaluating binary classification tasks. the Receiver Operating Characteristic (ROC) curve, which is a plot of the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings. FPR is defined as proportion of actual negative cases that are incorrectly identified as positive by the classifier. The threshold settings are the values that define if the prediction will be assigned to class 0 or class 1. For example, the classifier predict for an example value 0.252. For threshold 0.2 the prediction would be class 1 (because $0.252 > 0.2$), and for threshold 0.5 the prediction would be class 0 (because $0.252 < 0.5$). Because the evaluation of different thresholds, AUC is also useful with imbalanced class distribution. AUC values ranges from 0 to 1 and the higher value, the better classifier performance.

Balanced accuracy was calculated for train and test dataset. AUC was calculated for test dataset. The balanced accuracy on train dataset is presented in Figure 6.2. The balanced accuracy calculated on test dataset is presented in Figure 6.3. The AUC metrics are presented in Figure 6.4.

Analysing results presented in Figures 6.2, 6.3 and 6.4, we can see that Logistic Regression obtained the worst results. We can conclude that this method did not fit the task of forecasting product return. The Random Forest method had high value of Balance Accuracy on train dataset (Figure 6.2) but after the parameter tuning

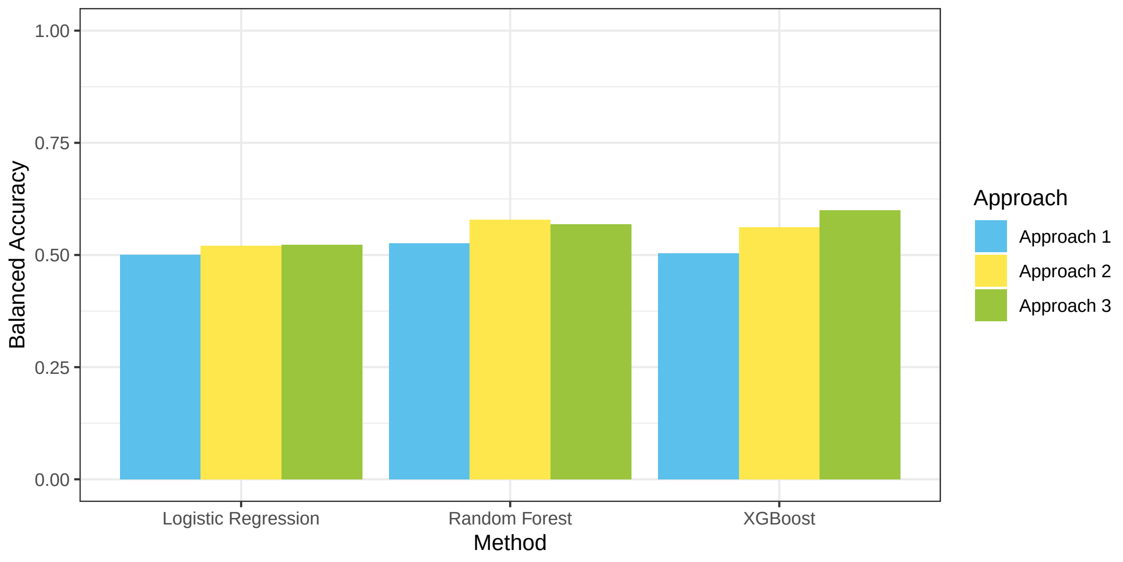


Figure 6.3: Balanced Accuracy calculated on test dataset for each method and each evaluated approach.

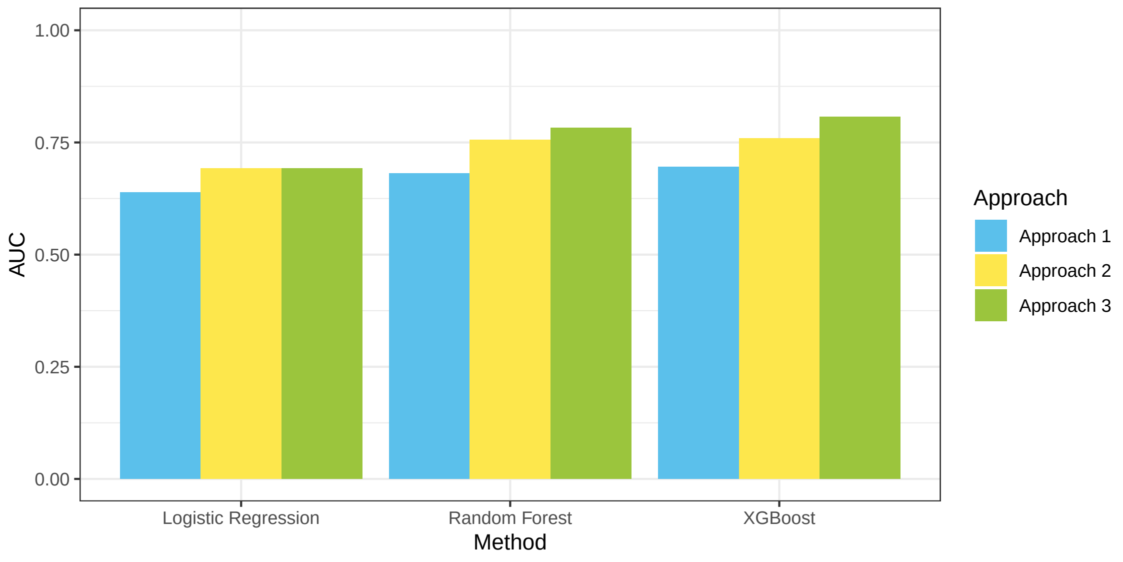


Figure 6.4: Area Under the Curve values for each method and each evaluated approach.

procedure this value drops significantly to 0.714. We can see that the parameter tuning for Random Forest has not had the desired effect. Also, even for method 1 and 2, the Balanced Accuracy for test dataset is relatively low comparing to results obtained on train dataset (Figure 6.3). The situation is different for the XGBoost method. The successive steps carried out (adding new conditional attributes, tuning parameters), caused that each successive approach improved the obtained classification results, on the training set as well as on the test set. It is visible on the plots presented Blanced Accuaracy on train and test dataset (Figures 6.2 and 6.3). Taking into account the results of Balanced Accuracy, the best result on test dataset was obtained for XGBoost for Approach number 3 with result of Balanced Accuracy equal 0.6. Analyzing the results of AUC, the best result was obtained also for XGboost for Approach number 3 (after feature engineering and parameter tuning) with the value of AUC equal 0.806.

6.4 Discussion

Predictive models in terms of the AUC measure achieved satisfactory results, with the best result of AUC = 0.806 for the XGBoost method after feature engineering and parameter tuning. Looking at the Balanced Accuracy results, it can be concluded that the proposed models have some limitations, because the results of the Balanced Accuracy measure are below the value of 0.6. However, it is worth noting that this result was obtained on the basis of a classification threshold of 0.5 (predictions below this value were assigned to class 0, and above – to class 1). It is possible that choosing a different cutoff threshold would have produced better results. This threshold, however, would have to be established with an expert with domain knowledge, as they could indicate what are the preferred specificity and sensitivity for this task. The AUC measure was also a metric used in the parameter tuning process, so it is worth focusing on this measure in the evaluation. After the study, it can be concluded that the linear regression method should not be considered for possible further research.

Another conclusion is that the feature engineering and parameter tuning performed gave good results, especially for the XGBoost method. It can be concluded that the addition of features describing the content of the shopping basket, including the purchase of many similar products, improved the results for each method.

The proposed predictive model could be improved, but this would require acquiring more attributes. It can be noted that we did not have much information about the products in question in the data – we only had information about color, size, price and product category. We didn't have information, about the special features of a given product, which could distinguish it from other products. There was also no information about the material, cut or quality of product. Such features could affect possible returns and customer dissatisfaction. However, these features are difficult to enter into the database, as they are often subjective and time-consuming to obtain. For example, a blouse made of 100 percent cotton may have a completely different feel than another blouse also made of 100 percent cotton. This can be caused, for example, by the weight of the material, which can affect the fact that one blouse will be fleshy in feel and the other will be almost see-through. Such characteristics are difficult to enter into the database, especially if the supplier does not provide such information.

It is possible that the use of graphical data, i.e. product images, would be better for returns analysis. A model based on images could catch product features that affect returns, which are difficult to express numerically. However, we did not have access to such data.

It is also worth noting that the classes used regarding returns were determined by a proprietary algorithm linking the purchase transaction to the return, rather than from actual data. This could also have introduced additional bias.

6.5 Conclusions

The chapter focuses on the problem of forecasting returns. More specifically it concentrates on forecasting whether a particular product from a particular purchase transaction will be returned. After conducting an analysis of the literature, it can be concluded that this is a possible research gap because as far as we know there aren't work that focused on this specific problem.

The achievement is the proposed algorithm for linking a return with a corresponding purchase transaction. This gives the possibility to prepare a training set in case there is no connection between these transactions in the retailer's database.

In the research, three methods were used to create predictive models: logistic regression, Random Forests and XGBoost. The process of improving the model through feature engineering and parameter tuning was presented. The study showed that the addition of features describing the shopping cart and the purchase of similar products positively affects the quality of the classifiers. The best AUC result was obtained for the XGBoost method with an AUC score = 0.806.

7 Summary

Demand forecasting will take place in almost every company. In order to optimise and automate processes, it is possible to use machine learning methods for the task.

In the thesis, the original breakdown of demand forecasting issues was presented. These issues have been divided by the time factor. Each task has also been described with the specific problems that may be associated with each of them. This breakdown can show a broader picture of the issues involved in demand forecasting. Then, attention was turned to selected issues and consideration was given to how machine learning methods can be applied to solve given problems.

The topic of promotion, and more specifically the analysis and forecasting of promotion effectiveness, was looked at first. This topic is important for optimal planning of future promotions, e.g. in shops. First, a set of indicators was proposed that can be used to evaluate promotions. The process of creating conditional attributes and a training set to train predictive models for each indicator is shown. The possibility of creating such predictive models for single products as well as for groups of products was considered, showing that forecasting from data for multiple products, is more applicable. It is then shown how the proposed predictive models can be used in a promotional recommendation system and to discover knowledge about the price sensitivity. Finally, a novel approach for analysing the quality of indicators using the original method for induction survival action rules is considered. The theory associated with survival action rules is shown and the proposed algorithm is described. A case-study of the use of the given algorithm to analyse changes that can be made to reduce the probability of an indicator falling below a given threshold is then shown. It is worth noting that the given approach could also be used to analyse other indicators whose value changes over time, not necessarily only for indicators describing promotion.

The next topic raised was the issue of splitting higher-level forecasts into lower-level forecasts. A higher-level forecast can be a forecast of a demand for a whole group of products proposed by an expert with domain knowledge. It can also be a forecast derived from another predictive learning model. A lower-level forecast is a forecast of demand for a particular output. Top-down forecasting can take place wherever products can be grouped together. It is relevant when a particular product has very low or infrequent sales, making it impossible to create a forecasting model for a specific product. This thesis chapter focuses on a comparison of selected forecasting algorithms. The possibility of using the *product types dissimilarity table* is presented, which allows the product models (nominal values) to be changed to the distances used, among others, in the considered approach with the KNN model. The process of data preparation and model optimisation is shown.

The next chapter presented the possibilities and advantages of using additional time series to improve the forecasting performance of the issue under consideration. First, the possibility of using analogous time series from other locations was shown. Different possibilities for grouping and adding information of similar time series were presented. For both sets studied, it was shown that creating forecasting models using all the data improves the results of the obtained forecasts. Next, a forecasting issue was looked at where time series information from the same location was added. This approach in itself was innovative for the field under study, namely energy and electricity forecasting. The literature review indicated that this approach was unique. It could be explored thanks to created a digital-twin model of a building. By obtaining data on electricity consumption from household appliances and feeding this into the forecasting model, improvements in the forecasting of overall energy consumption were achieved. An analysis of model errors and the possible application of this knowledge to reduce the observed current sources is also presented. The conclusion of this research is that adding data from other locations or the same location can significantly improve the quality of the forecasts. However, the way in which this information should be added can vary depending on the dataset and different options need to be verified.

The final case considered was the problem of forecasting returns. The literature review indicated that this was a possible research gap. The problem of forecasting returns had appeared in the literature, but in global terms (e.g. the number of total returns over a given period). To the best of our knowledge, previous studies have not looked at predicting the probability of a particular product being returned. Initially, a custom algorithm was proposed to combine the return transaction with the purchase transaction. The given algorithm was necessary in order to create a training set. As later results also showed, adding information about the purchase transaction and its characteristics improved the prediction results significantly. In the research for the given dataset, selected algorithms were considered, the optimisation process and feature engineering were shown. Good return forecast results were obtained. Good quality of classification was obtained (AUC = 0.806), confirming the effectiveness of the proposed process.

The thesis presented does not focus on the typical issue of time series forecasting in the sense that, given historical demand data, a forecast will be made for subsequent points in time. The aim of the given work was not to focus on a comparison of all known time series forecasting algorithms. Instead, the focus was on lesser-known demand forecasting issues.

All studies were carried out on real data. Most of them were from large companies with multiple branches. The data also often spanned several years and came from multiple locations and involved multiple products. The studies considered were dictated by R&D research projects, so it can be concluded that these are relevant to the business. Due to the cross-section of data used and issues addressed, it is hoped that the knowledge presented will be a valuable contribution to the machine learning research and business communities.

7.1 Future directions

Undoubtedly, this work does not exhaust the topic of demand forecasting using machine learning and domain knowledge.

The next development steps could certainly address the consideration of the remaining issues presented in Section 2.3 in the breakdown of demand forecasting tasks. Some of these, however, would require the acquisition of new data sets and consultation with experts with domain knowledge (e.g. for the problem of the product life cycle or the identification of reference products or shops).

Further research related to the proposed algorithm for the induction of survival action rules is an interesting line of development. The research could investigate the recommendation capabilities of the given algorithm for different business indicators.

It would also have been possible to explore in greater depth the very subject of making recommendations, i.e. the actions that could have been introduced to change the value of an indicator. On this topic, it could be interesting to try using a *counterfactuals* [162, 48]. One could compare the recommendations made by the action rules and counterfactuals.

Bibliography

[1] Nowy system rozliczania, tzw. net-billing. Available online: `https://www.gov.pl/web/klimat/nowy-system-rozliczania-tzw-net-billing`.

[2] Prophet. prophet, forecasting at scale. Available online: `https://facebook.github.io/prophet/`.

[3] *Introduction to the Logistic Regression Model*, rozdzia/l 1, strony 1–30. John Wiley & Sons, Ltd, 2000.

[4] *h2o: R Interface for H2O*, March 2020.

[5] Aamer, A., Eka Yani, L. P., Alan Priyatna, I. M. Data analytics in the supply chain management: Review of machine learning applications in demand forecasting. *Operations and Supply Chain Management: An International Journal*, 14(1):1–13, 2020.

[6] Adithya Ganesan, V., Divi, S., Moudhgalya, N. B., Sriharsha, U., Vijayaraghavan, V. Forecasting food sales in a multiplex using dynamic artificial neural networks. *Advances in Intelligent Systems and Computing*, wolumen 944, strony 69–80. Springer Verlag, 2020.

[7] Aghabozorgi, S., Seyed Shirkhorshidi, A., Ying Wah, T. Time-series clustering - A decade review. *Information Systems*, 53:16–38, may 2015.

[8] Alhussein, M., Aurangzeb, K., Haider, S. I. Hybrid cnn-lstm model for short-term individual household load forecasting. *IEEE Access*, 8:180544–180557, 2020.

[9] Ali, Ö. G., Sayin, S., van Woensel, T., Fransoo, J. SKU demand forecasting in the presence of promotions. *Expert Systems with Applications*, 36(10):12340–12348, dec 2009.

[10] Alkhraijah, M., Alowaifeer, M., Alsaleh, M., Alfaris, A., Molzahn, D. K. The effects of social distancing on electricity demand considering temperature dependency. *Energies*, 14(2), 2021.

[11] Almazrouee, A. I., Almeshal, A. M., Almutairi, A. S., Alenezi, M. R., Alhajeri, S. N. Long-term forecasting of electrical loads in kuwait using prophet and holt–winters models. *Applied Sciences*, 10(16), 2020.

[12] Amasyali, K., El-Gohary, N. M. A review of data-driven building energy consumption prediction studies. *Renewable and Sustainable Energy Reviews*, 81:1192–1205, 2018.

[13] Amber, K. P., Aslam, M. W., Mahmood, A., Kousar, A., Younis, M. Y., Akbar, B., Chaudhary, G. Q., Hussain, S. K. Energy consumption forecasting for university sector buildings. *Energies*, 10(10), 2017.

[14] Ambilkar, P., Dohale, V., Gunasekaran, A., Bilolikar, V. Product returns management: a comprehensive review and future research agenda. *International Journal of Production Research*, 60(12):3920–3944, 2022.

[15] Arkes, H. R. Overconfidence in judgmental forecasting. *Journal of Personality and Social Psychology*, 50(3):492–502, 1986.

[16] Au, K.-F., Choi, T.-M., Yu, Y. Fashion retail forecasting by evolutionary neural networks. *International Journal of Production Economics*, 114(2):615 – 630, 2008.

[17] Banerjee, N., Morton, A., Akartunalı, K. Passenger demand forecasting in scheduled transportation. *European Journal of Operational Research*, 286(3):797–810, 2020.

[18] Barua, L., Zou, B., Zhou, Y. Machine learning for international freight transportation management: A comprehensive review. *Research in Transportation Business & Management*, 34:100453, 2020.

[19] Beaudin, M., Zareipour, H. Home energy management systems: A review of modelling and complexity. *Renewable and Sustainable Energy Reviews*, 45:318–335, 2015.

[20] Beheshti-Kashi, S., Karimi, H. R., Thoben, K.-D., Lütjen, M., Teucke, M. A survey on retail sales forecasting and prediction in fashion markets. *Systems Science & Control Engineering*, 3(1):154–161, 2015.

[21] Benidis, K., Rangapuram, S. S., Flunkert, V., Wang, Y., Maddix, D., Turkmen, C., Gasthaus, J., Bohlke-Schneider, M., Salinas, D., Stella, L., Aubet, F.-X., Callot, L., Januschowski, T. Deep learning for time series forecasting: Tutorial and literature survey. *ACM Comput. Surv.*, 55(6), dec 2022.

[22] Bewick, V., Cheek, L., Ball, J. Statistics review 12: survival analysis. *Critical care*, 8(5):1–6, 2004.

[23] Bianco, V., Manca, O., Nardini, S. Linear regression models to forecast electricity consumption in italy. *Energy Sources, Part B: Economics, Planning, and Policy*, 8(1):86–93, 2013.

[24] Blachnik, M., Henzel, J. Estimating the performance indicators of promotion efficiency in fmcg retail", booktitle="neural information processing. strony 320–332, Cham, 2020. Springer International Publishing.

[25] Blattberg, R. C., Levin, A. Modelling the effectiveness and profitability of trade promotions. *Marketing Science*, 6(2):124–146, 1987.

[26] Borghini, E., Giannetti, C., Flynn, J., Todeschini, G. Data-driven energy storage scheduling to minimise peak demand on distribution systems with pv generation. *Energies*, 14(12), 2021.

[27] Bouktif, S., Fiaz, A., Ouni, A., Serhani, M. A. Multi-sequence lstm-rnn deep learning and metaheuristics for electric load forecasting. *Energies*, 13(2), 2020.

[28] Box, G. E., Jenkins, G. M. *Time series analysis: forecasting and control.* Holden-Day, 1970.

[29] Box, G. E., Jenkins, G. M., Reinsel, G. C., Ljung, G. M. *Time series analysis: forecasting and control.* John Wiley & Sons, 2015.

[30] Braulio-Gonzalo, M., Bovea, M. D., Jorge-Ortiz, A., Juan, P. Contribution of households' occupant profile in predictions of energy consumption in residential buildings: A statistical approach from mediterranean survey data. *Energy and Buildings*, 241:110939, 2021.

[31] Breiman, L. Random forests. *Machine learning*, 45(1):5–32, 2001.

[32] Breiman, L., Friedman, J., Stone, C. J., Olshen, R. A. *Classification and Regression Trees*. Chapman and Hall/CRC, 1984.

[33] Brownlee, J. *Deep Learning for Time Series Forecasting: Predict the Future with MLPs, CNNs and LSTMs in Python*. Machine Learning Mastery, 2018.

[34] Bu, S.-J., Cho, S.-B. Time series forecasting with multi-headed attention-based deep learning for residential energy consumption. *Energies*, 13(18), 2020.

[35] Budholiya, K., Shrivastava, S. K., Sharma, V. An optimized xgboost based diagnostic system for effective prediction of heart disease. *Journal of King Saud University - Computer and Information Sciences*, 34(7):4514–4523, 2022.

[36] Chen, C. Y., Lee, W. I., Kuo, H. M., Chen, C. W., Chen, K. H. The study of a forecasting sales model for fresh food. *Expert Systems with Applications*, 37(12):7696–7702, dec 2010.

[37] Chen, T., Guestrin, C. XGBoost: A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, wolumen KDD '16, strony 785–794. ACM, 2016.

[38] Chen, T., He, T., Benesty, M., Khotilovich, V., Tang, Y., Cho, H., Chen, K., Mitchell, R., Cano, I., Zhou, T., Li, M., Xie, J., Lin, M., Geng, Y., Li, Y. *xgboost: Extreme Gradient Boosting*, 2019.

[39] Chollet, F., i in. Keras. `https://keras.io`, 2015.

[40] Chu, C. W., Zhang, G. P. A comparative study of linear and nonlinear models for aggregate retail sales forecasting. *International Journal of Production Economics*, 86(3):217–231, dec 2003.

[41] Ciulla, G., D'Amico, A. Building energy performance forecasting: A multiple linear regression approach. *Applied Energy*, 253:113500, 2019.

[42] Cleveland, R. B., Cleveland, W. S., McRae, J. E., Terpenning, I. Stl: A seasonal-trend decomposition procedure based on loess. *Journal ofOfficial Statistics*, 6(1):3–73, 1990.

[43] Cohen, M. C., Leung, N. H. Z., Panchamgam, K., Perakis, G., Smith, A. The impact of linear optimization on promotion planning. *Operations Research*, 65(2):446–468, 2017.

[44] Cui, G., Wong, M. L., Lui, H. K. Machine learning for direct marketing response models: Bayesian networks with evolutionary programming. *Management Science*, 52(4):597–612, 2006.

[45] Cui, H., Rajagopalan, S., Ward, A. R. Predicting product return volume using machine learning methods. *European Journal of Operational Research*, 281(3):612–627, 2020.

[46] Czosnyka, M., Wnukowska, B., Karbowa, K. Electrical energy consumption and the energy market in poland during the covid-19 pandemic. *2020 Progress in Applied Electrical Engineering (PAEE)*, strony 1–5, 2020.

[47] Dairu, X., Shilong, Z. Machine learning model for sales forecasting by using xgboost. *2021 IEEE International Conference on Consumer Electronics and Computer Engineering (ICCECE)*, strony 480–483, 2021.

[48] Dandl, S., Molnar, C., Binder, M., Bischl, B. Multi-objective counterfactual explanations. Bäck, T., Preuss, M., Deutz, A., Wang, H., Doerr, C., Emmerich, M., Trautmann, H., redaktorzy, *Parallel Problem Solving from Nature – PPSN XVI*, strony 448–469, Cham, 2020. Springer International Publishing.

[49] Dardzinska, A. *Action rules mining*, wolumen 468. Springer, 2012.

[50] Deb, C., Zhang, F., Yang, J., Lee, S. E., Shah, K. W. A review on time series forecasting techniques for building energy consumption. *Renewable and Sustainable Energy Reviews*, 74(March):902–924, 2017.

[51] Demšar, J. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, 7:1–30, dec 2006.

[52] Doganis, P., Alexandridis, A., Patrinos, P., Sarimveis, H. Time series sales forecasting for short shelf-life food products based on artificial neural networks and evolutionary computing. *Journal of Food Engineering*, 75(2):196–204, jul 2006.

[53] du Preez, J., Witt, S. F. Univariate versus multivariate time series forecasting: an application to international tourism demand. *International Journal of Forecasting*, 19(3):435–451, 2003.

[54] Erdogdu, E. Electricity demand analysis using cointegration and arima modelling: A case study of turkey. *Energy Policy*, 35(2):1129–1146, 2007.

[55] Fan, C., Xiao, F., Wang, S. Development of prediction models for next-day building energy consumption and peak power demand using data mining techniques. *Applied Energy*, 127:1–10, 2014.

[56] Fildes, R., Goodwin, P., Önkal, D. Use and misuse of information in supply chain forecasting of promotion effects. *International Journal of Forecasting*, 35(1):144–156, jan 2019.

[57] Fox, J. *Applied regression analysis and generalized linear models.* Sage Publications, 2015.

[58] Friedman, J. H. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29:1189–1232, 2001.

[59] Friedman, M. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association*, 32(200):675–701, 1937.

[60] Fürnkranz, J., Gamberger, D., Lavrač, N. *Foundations of Rule Learning.* Cognitive Technologies. Springer Berlin Heidelberg, 2012.

[61] Fürnkranz, J., Gamberger, D., Lavrač, N. *Foundations of rule learning.* Springer Science & Business Media, 2012.

[62] Garcı́a, S., Herrera, F. An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons. *Journal of Machine Learning Research*, 9:2677–2694, 2008.

[63] Gardner, E. S. Exponential smoothing: The state of the art—part ii. *International Journal of Forecasting*, 22(4):637–666, 2006.

[64] Gardner Jr., E. S. Exponential smoothing: The state of the art. 4(October 1983):1–28, 1985.

[65] Goedhart, J., Haijema, R., Akkerman, R. Modelling the influence of returns for an omni-channel retailer. *European Journal of Operational Research*, 306(3):1248–1263, 2023.

[66] Goodwin, P., Wright, G. *Forecasting with judgment.* John Wiley & Sons, 2004.

[67] Hajek, P., Abedin, M. Z., Sivarajah, U. Fraud detection in mobile payment systems using an xgboost-based framework. *Information Systems Frontiers*, Paz. 2022.

[68] Hasan Shawon, M. M., Akter, S., Islam, M. K., Ahmed, S., Rahman, M. M. Forecasting pv panel output using prophet time series machine learning model. *2020 IEEE REGION 10 CONFERENCE (TENCON)*, strony 1141–1144, 2020.

[69] Haselbeck, F., Killinger, J., Menrad, K., Hannus, T., Grimm, D. G. Machine learning outperforms classical forecasting on horticultural sales predictions. *Machine Learning with Applications*, 7:100239, 2022.

[70] He, Y., Xu, Q., Wu, P. Omnichannel retail operations with refurbished consumer returns. *International Journal of Production Research*, 58(1):271–290, 2020.

[71] Henzel, J., Bularz, J., Sikora, M. Impact of time series clustering on fuel sales prediction results. Ganzha, M., Maciaszek, L., Paprzycki, M., Ślęzak, D., redaktorzy, *Position and Communication Papers of the 16th Conference on Computer Science and Intelligence Systems*, wolumen 26 serii *Annals of Computer Science and Information Systems*, strona 13–21. PTI, 2021.

[72] Henzel, J., Sikora, M. Gradient boosting and deep learning models approach to forecasting promotions efficiency in fmcg retail. Rutkowski, L., Scherer, R., Korytkowski, M., Pedrycz, W., Tadeusiewicz, R., Zurada, J. M., redaktorzy, *Artificial Intelligence and Soft Computing*, strony 336–345, Cham, 2020. Springer International Publishing.

[73] Henzel, J., Sikora, M. Gradient boosting application in forecasting of performance indicators values for measuring the efficiency of promotions in fmcg retail. Agarwal, S., Barrell, D. N., Solanki, V. K., redaktorzy, *Proceedings of the 2020 Federated Conference on Computer Science and Information Systems*, wolumen 21 serii *Annals of Computer Science and Information Systems*, strony 59–68. IEEE, 2020.

[74] Henzel, J., Wawrowski, Ł., Kubina, A., Sikora, M., Wróbel, Ł. Demand forecasting in the fashion business — an example of customized nearest neighbour and linear mixed model approaches. strony 61–65, 09 2022.

[75] Henzel, J., Wróbel, Ł., Fice, M., Sikora, M. Energy consumption forecasting for the digital-twin model of the building. *Energies*, 15(12), 2022.

[76] Hochreiter, S., Schmidhuber, J. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 11 1997.

[77] Hong, T., Gui, M., Baran, M. E., Willis, H. L. Modeling and forecasting hourly electric load by multiple linear regression with interactions. *IEEE PES General Meeting*, strony 1–8, 2010.

[78] Huang, T., Fildes, R., Soopramanien, D. The value of competitive information in forecasting FMCG retail product sales and the variable selection problem. *European Journal of Operational Research*, 237(2):738–748, sep 2014.

[79] Huang, Y., Xu, C., Ji, M., Xiang, W., He, D. Medical service demand forecasting using a hybrid model based on arima and self-adaptive filtering method. *BMC Medical Informatics and Decision Making*, 20(1):237, 2020.

[80] Hundman, K., Constantinou, V., Laporte, C., Colwell, I., Soderstrom, T. Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, strona 387–395, New York, NY, USA, 2018. Association for Computing Machinery.

[81] Hyndman, R. J., Athanasopoulos, G. *Forecasting: principles and practice.* OTexts, 2018.

[82] Iman, R. L., Davenport, J. M. Approximations of the critical region of the friedman statistic.

[83] Ingle, C., Bakliwal, D., Jain, J., Singh, P., Kale, P., Chhajed, V. Demand forecasting : Literature review on various methodologies. *2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, strony 1–7, 2021.

[84] Islek, I., Gunduz Oguducu, S. A decision support system for demand forecasting based on classifier ensemble. *Communication Papers of the 2017 Federated Conference on Computer Science and Information Systems*, strony 35–41, 2017.

[85] Iwafune, Y., Yagita, Y., Ikegami, T., Ogimoto, K. Short-term forecasting of residential building load for distributed energy management. *2014 IEEE International Energy Conference (ENERGYCON)*, strony 1197–1204, 2014.

[86] Jadwiszczak, P., Jurasz, J., Kaźmierczak, B., Niemierka, E., Zheng, W. Factors shaping a/w heat pumps co_2 emissions—evidence from poland. *Energies*, 14(6), 2021.

[87] Jain, R. K., Smith, K. M., Culligan, P. J., Taylor, J. E. Forecasting energy consumption of multi-family residential buildings using support vector regression: Investigating the impact of temporal and spatial monitoring granularity on performance accuracy. *Applied Energy*, 123:168–178, 2014.

[88] Javed, A., Lee, B. S., Rizzo, D. M. A benchmark study on time series clustering. *Machine Learning with Applications*, 1:100001, 2020.

[89] Jetcheva, J. G., Majidpour, M., Chen, W.-P. Neural network model ensembles for building-level electricity load forecasts. *Energy and Buildings*, 84:214–223, 2014.

[90] Ji, S., Wang, X., Zhao, W., Guo, D. An application of a three-stage xgboost-based model to sales forecasting of a cross-border e-commerce enterprise. *Mathematical Problems in Engineering*, 2019, Sep 2019.

[91] Jiménez-Pérez, P. F., Mora-López, L. Modeling and forecasting hourly global solar radiation using clustering and classification techniques. *Solar Energy*, 135:682–691, oct 2016.

[92] Jovanović, R. Ž., Sretenović, A. A., Živković, B. D. Ensemble of various neural networks for prediction of heating energy consumption. *Energy and Buildings*, 94:189–199, 2015.

[93] K U, J., Kovoor, B. Deterministic weather forecasting models based on intelligent predictors: A survey. *Journal of King Saud University - Computer and Information Sciences*, 34, 09 2020.

[94] Kaczmarek, K., Hryniewicz, O. Linguistic knowledge about temporal data in bayesian linear regression model to support forecasting of time series. *2013 Federated Conference on Computer Science and Information Systems, FedCSIS 2013*, strony 651–654, 2013.

[95] Khajavi, S. H., Motlagh, N. H., Jaribion, A., Werner, L. C., Holmström, J. Digital twin: Vision, benefits, boundaries, and creation for buildings. *IEEE Access*, 7:147406–147419, 2019.

[96] Kim, J.-Y., Cho, S.-B. Electric energy consumption prediction by deep learning with state explainable autoencoder. *Energies*, 12(4), 2019.

[97] Koltsaklis, N., Panapakidis, I. P., Pozo, D., Christoforidis, G. C. A prosumer model based on smart home energy management and forecasting techniques. *Energies*, 14(6):1–32, 2021.

[98] Kordos, M., Blachnik, M. Instance selection with neural networks for regression problems. *LNCS*, 7553:263–270, 2012.

[99] Kordos, M., Strzempa, D., Blachnik, M. Do we need whatever more than k-nn? *LNCS*, 6113:414–421, 2010.

[100] Kozielski, M., Henzel, J., Wróbel, Ł., Łaskarzewski, Z., Sikora, M. A sensor data-driven decision support system for liquefied petroleum gas suppliers. *Applied Sciences*, 11(8), 2021.

[101] Krishna, A., Akhilesh, V., Aich, A., Hegde, C. Sales-forecasting of retail stores using machine learning techniques. *Sales-forecasting of Retail Stores using Machine Learning Techniques*, strony 160–166. IEEE, 2018.

[102] L., Kaplan, E., Meier, P. Nonparametric estimation from incomplete observations. *Journal of American Statistical Association*, 53:457–481, 06 1958.

[103] Lara-Benítez, P., Carranza-García, M., Riquelme, J. C. An experimental review on deep learning architectures for time series forecasting. *International Journal of Neural Systems*, 31(03):2130001, 2021.

[104] Lawrence, M., Goodwin, P., O'Connor, M., Önkal, D. Judgmental forecasting: A review of progress over the last 25 years. *International Journal of Forecasting*, 22(3):493–518, 2006.

[105] Li, H. Multivariate time series clustering based on common principal component analysis. *Neurocomputing*, 349:239–247, 2019.

[106] Li, Y., Zhu, Z., Kong, D., Han, H., Zhao, Y. Ea-lstm: Evolutionary attention-based lstm for time series prediction. *Knowledge-Based Systems*, 181:104785, 2019.

[107] Li, Z., Liu, F., Yang, W., Peng, S., Zhou, J. A survey of convolutional neural networks: Analysis, applications, and prospects. *IEEE Transactions on Neural Networks and Learning Systems*, 33(12):6999–7019, 2022.

[108] Liu, F., Cai, M., Wang, L., Lu, Y. An ensemble model based on adaptive noise reducer and over-fitting prevention lstm for multivariate time series forecasting. *IEEE Access*, 7:26102–26115, 2019.

[109] Liu, J., Wang, S., Wei, N., Chen, X., Xie, H., Wang, J. Natural gas consumption forecasting: A discussion on forecasting history and future challenges. *Journal of Natural Gas Science and Engineering*, 90:103930, 2021.

[110] Liu, Y., Feng, G., Chin, K.-S., Sun, S., Wang, S. Daily tourism demand forecasting: the impact of complex seasonal patterns and holiday effects. *Current Issues in Tourism*, 26(10):1573–1592, 2023.

[111] Lysenko-Ryba, K., Zimon, D. Customer behavioral reactions to negative experiences during the product return. *Sustainability*, 13(2), 2021.

[112] Makridakis, S. The art and science of forecasting An assessment and future directions. *International Journal of Forecasting*, 2(1):15–39, 1986.

[113] Marino, D. L., Amarasinghe, K., Manic, M. Building energy load forecasting using deep neural networks. *IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society*, strony 7046–7051, 2016.

[114] Markakis, E. K., Nikoloudakis, Y., Lapidaki, K., Fiorentzis, K., Karapidakis, E. Unification of edge energy grids for empowering small energy producers. *Sustainability*, 13(15), 2021.

[115] McCullagh, P., Nelder, J. A. *Generalized linear models*, wolumen 37. CRC press, 1989.

[116] Motlagh, O., Berry, A., O'Neil, L. Clustering of residential electricity customers using load time series. *Applied Energy*, 237:11–24, 2019.

[117] Murtagh, F. Multilayer perceptrons for classification and regression. *Neurocomputing*, 2(5):183–197, 1991.

[118] Nemenyi, P. B. *Distribution-Free Multiple Comparisons.* Praca doktorska, Princeton University, 1963.

[119] Nepal, B., Yamaha, M., Yokoe, A., Yamaji, T. Electricity load forecasting using clustering and ARIMA model for energy management in buildings. *Japan Architectural Review*, 3(1):62–76, jan 2020.

[120] Nguyen, D. H., de Leeuw, S., Dullaert, W. E. Consumer behaviour and order fulfilment in online retailing: A systematic review. *International Journal of Management Reviews*, 20(2):255–276, 2018.

[121] Niknam, A., Zare, H. K., Hosseininasab, H., Mostafaeipour, A., Herrera, M. A critical review of short-term water demand forecasting tools — what method should i use? *Sustainability*, 14(9), 2022.

[122] Niu, Y. Walmart sales forecasting using xgboost algorithm and feature engineering. *2020 International Conference on Big Data & Artificial Intelligence & Software Engineering (ICBASE)*, strony 458–461, 2020.

[123] Nti, I. K., Teimeh, M., Nyarko-Boateng, O., Adekoya, A. F. Electricity load forecasting: a systematic review. *Journal of Electrical Systems and Information Technology*, 7(1):13, 2020.

[124] O'Dwyer, E., Pan, I., Charlesworth, R., Butler, S., Shah, N. Integration of an energy management tool and digital twin for coordination and control of multi-vector smart energy systems. *Sustainable Cities and Society*, 62:102412, 2020.

[125] Ogunleye, A., Wang, Q.-G. Xgboost model for chronic kidney disease diagnosis. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 17(6):2131–2140, 2020.

[126] Ozcanli, A. K., Yaprakdal, F., Baysal, M. Deep learning methods and applications for electrical power systems: A comprehensive review. *International Journal of Energy Research*, 44(9):7136–7157, 2020.

[127] Parsa, A. B., Movahedi, A., Taghipour, H., Derrible, S., Mohammadian, A. K. Toward safer highways, application of xgboost and shap for real-time accident detection and feature analysis. *Accident Analysis & Prevention*, 136:105405, 2020.

[128] Paul Wright, S. Adjusted p-values for simultaneous inference. *Biometrics*, 48:1005–1013, 1992.

[129] Pumperla, M. Hyperas. `https://maxpumperla.com/hyperas/`, 2016.

[130] Reinhardt, H., Bergmann, J.-P., Münnich, M., Rein, D., Putz, M. A survey on modeling and forecasting the energy consumption in discrete manufacturing. *Procedia CIRP*, 90:443–448, 2020.

[131] Robertson, T. S., Hamilton, R., Jap, S. D. Many (un)happy returns? the changing nature of retail product returns and future research directions. *Journal of Retailing*, 96(2):172–177, 2020.

[132] Román-Portabales, A., López-Nores, M., Pazos-Arias, J. J. Systematic review of electricity demand forecast using ann-based machine learning algorithms. *Sensors*, 21(13), 2021.

[133] Saha, P., Gudheniya, N., Mitra, R., Das, D., Narayana, S., Tiwari, M. K. Demand forecasting of a multinational retail company using deep learning frameworks. *IFAC-PapersOnLine*, 55(10):395–399, 2022.

[134] Sahoo, N., Dellarocas, C., Srinivasan, S. The impact of online product reviews on product returns. *Information Systems Research*, 29(3):723–738, 2018.

[135] Sanders, N. R., Ritzman, L. P. *Judgmental Adjustment of Statistical Forecasts*, strony 405–416. Springer US, Boston, MA, 2001.

[136] Sepehr, M., Eghtedaei, R., Toolabimoghadam, A., Noorollahi, Y., Mohammadi, M. Modeling the electrical energy consumption profile for residential buildings in Iran. *Sustainable Cities and Society*, 41(November 2017):481–489, 2018.

[137] Seyedan, M., Mafakheri, F. Predictive big data analytics for supply chain demand forecasting: methods, applications, and research opportunities. *Journal of Big Data*, 7(1):53, 2020.

[138] Sfetsos, A., Coonick, A. Univariate and multivariate forecasting of hourly solar radiation with artificial intelligence techniques. *Solar Energy*, 68(2):169–178, 2000.

[139] Sharma, V., Cortes, A., Cali, U. Use of Forecasting in Energy Storage Applications: A Review. *IEEE Access*, 9:114690–114704, 2021.

[140] Sikora, M. Wybrane metody oceny i przycinania reguł decyzyjnych. *Studia Informatica*, 33(3B):5–331.

[141] Singh, K., Booma, P. M., Eaganathan, U. E-commerce system for sale prediction using machine learning technique. *Journal of Physics: Conference Series*, 1712(1):012042, dec 2020.

[142] Song, S. Y., Leng, H. Modeling the household electricity usage behavior and energy-saving management in severely cold regions. *Energies*, 13(21), 2020.

[143] Sosnowski, Ł., Szymusik, I., Penza, T. Network of Fuzzy Comparators for Ovulation Window Prediction. Lesot, M.-J., Vieira, S., Reformat, M. Z., Carvalho, J. P., Wilbik, A., Bouchon-Meunier, B., Yager, R. R., redaktorzy, *Information Processing and Management of Uncertainty in Knowledge-Based Systems*, strony 800–813, Cham, 2020. Springer International Publishing.

[144] Stevenson, M. An introduction to survival analysis. EpiCentre, IVABS, Massey University, 2007.

[145] Sun, Z.-L., Choi, T.-M., Au, K.-F., Yu, Y. Sales forecasting using extreme learning machine with applications in fashion retailing. *Decision Support Systems*, 46(1):411–419, 2008.

[146] Sundström, M., Hjelm-Lidholm, S., Radon, A. Clicking the boredom away – exploring impulse fashion buying behavior online. *Journal of Retailing and Consumer Services*, 47:150–156, 2019.

[147] Swaminathan, K., Venkitasubramony, R. Demand forecasting for fashion products: A systematic review. *International Journal of Forecasting*, 2023.

[148] Tarallo, E., Akabane, G. K., Shimabukuro, C. I., Mello, J., Amancio, D. Machine learning in predicting demand for fast-moving consumer goods: An exploratory research. *IFAC-PapersOnLine*, 52(13):737–742, 2019.

[149] Taylor, S. J., Letham, B. Forecasting at scale. *The American Statistician*, 72(1):37–45, 2018.

[150] Thomassey, S., Fiordaliso, A. A hybrid sales forecasting system based on clustering and decision trees. *Decision Support Systems*, 42(1):408–421, oct 2006.

[151] Thomassey, S., Fiordaliso, A. A hybrid sales forecasting system based on clustering and decision trees. *Decision Support Systems*, 42(1):408–421, oct 2006.

[152] Torlay, L., Perrone-Bertolotti, M., Thomas, E., Baciu, M. Machine learning–xgboost analysis of language networks to classify patients with epilepsy. *Brain Informatics*, 4(3):159–169, Wrze. 2017.

[153] Trapero, J. R., Kourentzes, N., Fildes, R. On the identification of sales forecasting models in the presence of promotions. *Journal of the Operational Research Society*, 66(2):299–307, 2015.

[154] Trizoglou, P., Liu, X., Lin, Z. Fault detection by an ensemble framework of extreme gradient boosting (xgboost) in the operation of offshore wind turbines. *Renewable Energy*, 179:945–962, 2021.

[155] Tso, G. K., Yau, K. K. Predicting electricity energy consumption: A comparison of regression analysis, decision tree and neural networks. *Energy*, 32(9):1761–1768, 2007.

[156] Tsoumakas, G. A survey of machine learning techniques for food sales prediction. *Artificial Intelligence Review*, 52(1):441–447, 2019.

[157] Turgut, Y., Erdem, M. Forecasting of retail produce sales based on xgboost algorithm. Calisir, F., redaktor, *Industrial Engineering in the Internet-of-Things World*, strony 27–43, Cham, 2022. Springer International Publishing.

[158] Van Donselaar, K. H., Peters, J., De Jong, A., Broekmeulen, R. Analysis and forecasting of demand during promotions for perishable items. *International Journal of Production Economics*, 172:65–75, feb 2016.

[159] Vartholomaios, A., Karlos, S., Kouloumpris, E., Tsoumakas, G. *Short-Term Renewable Energy Forecasting in Greece Using Prophet Decomposition and Tree-Based Ensembles*, strony 227–238. 09 2021.

[160] Vrindavanam, J., Babu, T., Gandiboina, H., Jayadev, G. A comparative analysis of machine learning algorithms for agricultural drought forecasting. *2022 3rd International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT)*, strony 1–6, 2022.

[161] Wachter, P., Widmer, T., Klein, A. Predicting automotive sales using pre-purchase online search data. *Proceedings of the 2019 Federated Conference on Computer Science and Information Systems*, strony 569–577, 2019.

[162] Wachter, S., Mittelstadt, B. D., Russell, C. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Cybersecurity*, 2017.

[163] Wahedi, H., Wrona, K., Heltoft, M., Saleh, S., Knudsen, T. R., Bendixen, U., Nielsen, I., Saha, S., Borup, G. S. Improving accuracy of time series forecasting by applying an arima-ann hybrid model. Kim, D. Y., von Cieminski, G., Romero, D., redaktorzy, *Advances in Production Management Systems. Smart Manufacturing and Logistics Systems: Turning Ideas into Action*, strony 3–10, Cham, 2022. Springer Nature Switzerland.

[164] Walther, J., Weigold, M. A systematic review on predicting and forecasting the electrical energy consumption in the manufacturing industry. *Energies*, 14(4), 2021.

[165] Wan, R., Mei, S., Wang, J., Liu, M., Yang, F. Multivariate temporal convolutional network: A deep neural networks approach for multivariate time series forecasting. *Electronics*, 8(8), 2019.

[166] Wang, Y., Wang, J., Zhao, G., Dong, Y. Application of residual modification approach in seasonal arima for electricity demand forecasting: A case study of china. *Energy Policy*, 48:284–294, 2012.

[167] Wei, H., Zeng, Q. Research on sales forecast based on xgboost-lstm algorithm model. *Journal of Physics: Conference Series*, 1754(1):012191, feb 2021.

[168] Witten, I. H., Frank, E., Hall, M. A. Chapter 7 - data transformations. Witten, I. H., Frank, E., Hall, M. A., redaktorzy, *Data Mining: Practical Machine Learning Tools and Techniques (Third Edition)*, The Morgan Kaufmann Series in Data Management Systems, strony 305–349. Morgan Kaufmann, Boston, wydanie third edition, 2011.

[169] Wohlrab, L., Fürnkranz, J. A review and comparison of strategies for handling missing values in separate-and-conquer rule learning. *Journal of Intelligent Information Systems*, 36(1):73–98, 2011.

[170] Wróbel, Ł., Gudyś, A., Sikora, M. Learning rule sets from survival data. *BMC Bioinformatics*, 18(285):1–13, 2017.

[171] Wu, Z., Pan, S., Long, G., Jiang, J., Chang, X., Zhang, C. Connecting the dots: Multivariate time series forecasting with graph neural networks. *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, strona 753–763, New York, NY, USA, 2020. Association for Computing Machinery.

[172] Xia, M., Zhang, Y., Weng, L., Ye, X. Fashion retailing forecasting based on extreme learning machine with adaptive metrics of inputs. *Knowledge-Based Systems*, 36:253–259, dec 2012.

[173] Xiao, Y., Yin, H., Zhang, Y., Qi, H., Zhang, Y., Liu, Z. A dual-stage attention-based conv-lstm network for spatio-temporal correlation and multivariate time series prediction. *International Journal of Intelligent Systems*, 36(5):2036–2057, 2021.

[174] Yan, K., Li, W., Ji, Z., Qi, M., Du, Y. A hybrid lstm neural network for energy consumption forecasting of individual households. *IEEE Access*, 7:157633–157642, 2019.

[175] Yu, Y., Choi, T.-M., Hui, C.-L. An intelligent fast sales forecasting model for fashion products. *Expert Systems with Applications*, 38(6):7373–7379, jun 2011.

[176] Zhang, J., Wedel, M. The effectiveness of customized promotions in online and offline stores. *Journal of Marketing Research*, 46(2):190–206, 2009.

[177] Zhang, R., Li, B., Jiao, B. Application of xgboost algorithm in bearing fault diagnosis. *IOP Conference Series: Materials Science and Engineering*, 490(7):072062, apr 2019.

[178] Zhang, T., Song, S., Li, S., Ma, L., Pan, S., Han, L. Research on gas concentration prediction models based on lstm multidimensional time series. *Energies*, 12(1), 2019.

[179] Zhang, Y., Luo, L., Yang, J., Liu, D., Kong, R., Feng, Y. A hybrid arima-svr approach for forecasting emergency patient flow. *Journal of Ambient Intelligence and Humanized Computing*, 10(8):3315–3323, 2019.

[180] Zhang, Y., Tong, J., Wang, Z., Gao, F. Customer transaction fraud detection using xgboost model. *2020 International Conference on Computer Engineering and Application (ICCEA)*, strony 554–558, 2020.

[181] Zhou, T., Ma, Z., xue wang, Wen, Q., Sun, L., Yao, T., Yin, W., Jin, R. FiLM: Frequency improved legendre memory model for long-term time series forecasting. Oh, A. H., Agarwal, A., Belgrave, D., Cho, K., redaktorzy, *Advances in Neural Information Processing Systems*, 2022.

[182] Zielińska-Sitkiewicz, M., Chrzanowska, M., Furmańczyk, K., Paczutkowski, K. Analysis of Electricity Consumption in Poland Using Prediction Models and Neural Networks. *Energies*, 14(20), 2021.