Politechnika Śląska
Wydział Automatyki, Elektroniki i Informatyki

Rozprawa Doktorska

Wykrywanie zmian danych wejściowych będących celowymi modyfikacjami ukierunkowanymi na wpływanie na wyniki działania modeli uczenia maszynowego

Piotr Biczyk

Promotor: dr hab. Marek Sikora, prof. Pol. Śl.

Gliwice 2024

Spis treści

1 Cele i Opis pracy

Poniższa praca powstała jako przyczynek do zagadnienia zapewnienia bezpieczeństwa systemom informatycznym zawierającym wbudowane mechanizmy uczenia maszynowego.

1.1 Otoczenie i rys historyczny

Rozwój technologii, od zarania dziejów, sprowadza się do dwóch przeplatających się procesów - otwierania nowych możliwości (poznawczych, produkcyjnych, etc.) oraz do umożliwienia, dzięki technologii, automatyzacji i optymalizacji czynności wykonywanych przez człowieka. Patrząc na obszar automatyzacji i optymalizacji, widać wyraźną różnicę pomiędzy rozwojem cywilizacji technicznej obserwowanym do połowy XX wieku a tym, który nastąpił w kolejnych dekadach. Tą różnicą jest zakres przedmiotowy automatyzacji. W ujęciu klasycznym automatyzacja i optymalizacja dotyczyła czynności fizycznych (silnik parowy zastępował konia pociągowego) lub wsparcia procesów podejmowania decyzji (człowiek dostawał nowe lepsze dane do podjęcia decyzji). Wprowadzenie i rozpowszechnienie komputerów umożliwiło automatyzację procesów decyzyjnych samych w sobie.

Jako stan bazowy można przyjąć okres, w którym decyzje biznesowe podejmowane były głównie na podstawie intuicji i osobistych doświadczeń osób zarządzających. Z czasem, w miarę rozwoju metod ilościowych, zaczęto wprowadzać proste algorytmy mające na celu strukturyzowanie i usprawnianie metod decyzyjnych. Te wczesne podejścia były reaktywne i rzadko opierały się na systematycznie zbieranych i przetwarzanych danych. Modelowanie matematyczne i analiza statystyczna umożliwiły tworzenie bardziej złożonych systemów decyzyjnych, oraz przetwarzanie dużych ilości danych i identyfikowanie w nich wzorców niewidocznych dla ludzkiego oka. Jednak dopiero postępy w informatyce i rosnąca dostępność mocy obliczeniowej pozwoliły na praktyczne zastosowanie technologii w automatyzacji podejmowania decyzji w procesach biznesowych. Pierwsze prace nad systemami wsparcia decyzji (DSS - De-

cision Support Systems) miały miejsce już w latach 60-tych XX wieku, skutkując znanymi praktycznymi implementacjami systemów eksperckich takich jak MYCIN (system diagnozowania chorób zakaźnych) czy DENDRAL (identyfikacja cząstek organicznych) [48]. Jednak dopiero rozkwit technik obliczeniowych oraz zdolności do ich wykonywania jaki wystąpił po roku 2010, umożliwił praktyczne zastosowanie mechanizmów automatyzacji podejmowania decyzji bazujących na uczeniu maszynowym. Wraz z popularnością i szerokim spektrum zastosowań technik uczenia maszynowego, pojawiły się, zarówno w rozważaniach teoretycznych jak i w świecie rzeczywistym, wyspecjalizowane techniki ataków na modele wytrenowane za pomocą metod maszynowego uczenia. Ataki mogą mieć różne cele - m.in. zaburzenie poprawnego działania modelu (np. poprzez zwiększenie liczby niepoprawnych decyzji generowanych przez model), obniżenie wiarygodności decyzji podejmowanych przez model, odtworzenie zbioru treningowego, odtworzenie modelu, etc. Ich wspólną cechą jest wykorzystanie specyficznych właściwości modeli i technik uczenia maszynowego. Ataki takie nazywamy atakami adwersaryjnymi (ang. Adversarial Machine Learning attacks - AML) [23].

1.2 Badania nad atakami adwersaryjnymi

Badania nad atakami adwersaryjnymi na modele uzyskane metodami uczenia maszynowego (ozn. modele ML - Machine Learning) rozpoczęły się od odkrycia przykładów adwersaryjnych (przykładów zakłócających, wprowadzających w błąd) w modelach klasyfikacji obrazów, w których niewielkie, starannie zaprojektowane zakłócenia w danych wejściowych mogą prowadzić do błędnej klasyfikacji. Przykład adwersaryjny zawiera celowo zmodyfikowane dane wejściowe. Wprowadzone zmiany są zazwyczaj niezauważalne dla człowieka, lecz wystarczają by wykorzystać podatności w granicach decyzyjnych modelu ML, prowadząc do nieprawidłowych wyników. Szegedy i in. [74] jako jedni z pierwszych udokumentowali to zjawisko, ujawniając, że sieci neuronowe są wysoce podatne na subtelne modyfikacje. Spostrzeżenie to wywołało duże zainteresowanie, a szerokie stosowanie sieci neuronowych spowodowało konieczność zrozumienia i przeciwdziałaniem tym podatnościom [3]. Podatność na ataki była szczególnie niepokojąca w kontekście praktycznych zastosowań modeli uczenia maszynowego, takich jak autonomiczne sterowanie pojazdami czy rozpoznawanie twarzy, gdzie zaburzone dane mogą prowadzić do błędów decyzyjnych o poważnych konsekwencjach — na przykład sprawiając, że samochód autonomiczny błędnie od-

czytuje znaki drogowe lub że system bezpieczeństwa nie rozpoznaje autoryzowanych osób [10].

Przykładem metody ataku jest metoda szybkiego gradientu (Fast Gradient Sign Method - FGSM) zaproponowaną przez Goodfellow'a [27]. Metoda wykorzystuje optymalizację gradientową do efektywnego generowania przykładów adwersaryjnych. Ataki prowadzone z wykorzystaniem optymalizacji gradientowej okazały się skuteczne nawet przy minimalnych modyfikacjach danych wejściowych [43]. Prace nad metodami ataków na modele ML realizowane są w dwóch głównych scenariuszach: *białej skrzynki* (atakujący zna w pełni architekturę i parametry modelu) oraz *czarnej skrzynki* (dostęp atakującego do atakowanego modelu jest ograniczony, zazwyczaj możliwe jest jedynie wykonywanie zapytań do modelu). Ważnym krokiem w rozwoju badań nad atakami adwersaryjnymi było zbadanie zakresu przenoszalności (transferowalności - analogia do transfer learning) ataków. Papernot i in. [60] wykazali, że przykłady adwersaryjne stworzone dla jednego modelu mogą oszukać inne modele wytrenowane na podobnych danych. Koncepcja ta umożliwia ataki nawet w środowiskach o ograniczonym dostępie, takich jak chmurowe usługi uczenia maszynowego [37].

Wymienione powyżej badania i metody skupiły się na badaniu ataków na modele służące klasyfikacji danych obrazowych, pomijając specyfikę modeli uczenia maszynowego operujących na danych tabelarycznych, powszechnie stosowanych w zastosowaniach takich jak finanse, opieka zdrowotna czy bezpieczeństwo. Dane tabelaryczne, w przeciwieństwie do obrazów czy tekstów, charakteryzują się specyficznymi strukturami i ograniczeniami, takimi jak różnorodność typów danych (m.in. symboliczna, numeryczne) oraz występująca często zależność pomiędzy zmiennymi opisującymi dane. Cechy te sprawiają, że tworzenie trudno wykrywalnych przykładów adwersaryjnych dla danych tabelarycznych jest bardziej wymagające, gdyż zakłócenia muszą pozostawać zgodne z rzeczywistymi ograniczeniami i zależnościami występującymi w danych. W tego rodzaju atakach konieczne jest stosowanie strategii uwzględniających specyficzne cechy danych i ich kontekstu, aby wprowadzone modyfikacje były wiarygodne i trudne do wykrycia [44]. Realistyczne ataki na dane tabelaryczne wymagają podejścia uwzględniającego specyficzne cechy poszczególnych zmiennych, zarówno numerycznych, jak i kategorycznych, oraz zachowania spójności z oczekiwanymi ograniczeniami wartości. Ataki uwzględniające ograniczenia wartości atrybutów lub zmiany wartości wybranych grup atrybutów, są kluczowe dla ukrycia faktu ataku - w danych tabelarycznych nawet drobne zmiany muszą mieścić się w realistycznych granicach [64]. Przykładowo, w kontekście finansów modyfikacje mogą koncentrować

się na krytycznych zmiennych, takich jak kwoty transakcji czy częstotliwość operacji, z zachowaniem cech statystycznych tych zmiennych, co pozwala na oszukanie modeli wykrywających oszustwa finansowe.

Przytoczone powyżej przykłady unaoczniają ryzyko jakie ataki adwersaryjne stanowią dla systemów komputerowych korzystających z modeli ML. Dotyczy to w szczególności systemów stosowanych w istotnych dziedzinach życia. Można tu wymienić takie obszary jak bezpieczeństwo drogowe (ataki na modele rozpoznawania znaków drogowych [22]), cyberbezpieczeństwo (oszukiwanie systemów IDS [66], kamuflaż złośliwego oprogramowania [19]), służbę zdrowia (modyfikacja danych medycznych [23]) czy finanse (ukrywanie nadużyć [39, 44]).

Warto przy tym zwrócić uwagę na nierównomierne rozłożenie badań dotyczących tworzenia i analizy właściwości metod ataków na modele przetwarzające obrazy oraz modele przetwarzające dane tabelaryczne. Dla przykładu przeszukanie bazy arXiv [1] daje jedenastokrotną różnicę w liczbie publikacji na korzyść danych obrazowych. Dla frazy "adversarial machine learning image data" znaleziono 1744 pozycje, a dla frazy "adversarial machine learning tabular data" jedynie 128[2].

1.3 Motywacja

Bezpośrednim impulsem do powstania niniejszej pracy były doświadczenia autora związane z zapewnieniem bezpieczeństwa w systemach automatyzujących procesy biznesowe i wykorzystujących mechanizmy uczenia maszynowego jako czynniki decyzyjne. Z doświadczeń tych wynika obserwacja powszechnego występowania luki polegającej na nieuwzględnianiu przy projektowaniu zabezpieczeń tych systemów specyficznych cech modeli uczenia maszynowego. W szczególności nieuwzględnieniu podatności modeli uczenia maszynowego na ataki typu adwersaryjnego. Dotyczy to zwłaszcza ataków, w których strona atakująca zaburza w sposób celowy dane przekazywane do modelu ML, w taki sposób aby zaburzenie to było jak najmniejsze, ale powodowało jak największy skutek.

Przez jak najmniejsze zaburzenie rozumie się zaburzenie danych prowadzone w sposób utrudniający ich wykrycie, często poniżej progu detekcji przez człowieka niewspartego dodatkowymi narzędziami. Przez jak największy skutek rozumie się osiągnięcie konkretnego celu ataku np. błędną klasyfikację danych. W szczegól-

[1] https://arxiv.org/
[2] Stan na 07.11.2024.

ności błędną klasyfikację polegającą na wskazaniu konkretnej klasy wymuszonej (preferowanej) przez metodę ataku.

Dodatkową motywacją podjęcia badań opisanych w niniejszej rozprawie były:

- Fakt, że większość prac dotycząca ataków adwersaryjnych opisuje metody ataków co nie przekłada się bezpośrednio na metody ich wykrywania.
- Brak rozwiniętych metod wykrywania ataków adwersaryjnych działających w paradygmacie *czarnej skrzynki* (braku dostępu do badanego/diagnozowanego modelu ML).
- Skupienie prac badawczych poświęconych zagadnieniom ataków adwersaryjnych na modelach przetwarzających obrazy przy niedoreprezentacji badań nad atakami na modele ML trenowane na danych tabelarycznych.

1.4 Cele i założenia pracy

1.4.1 Cel główny

Głównym celem pracy jest stworzenie, zweryfikowanie i implementacja metody wykrywania ataków adwersaryjnych na modele ML trenowane na danych tabelarycznych i pełniące rolę systemów klasyfikacyjnych. W celu zapewnienia uniwersalności, metoda ma działać w reżimie *czarnej skrzynki* tzn. bez dostępu do modelu poddawanego diagnostyce, w szczególności bez dostępu do jego architektury oraz parametrów. Działanie metody powinno opierać się na obserwacji decyzji podejmowanych przez diagnozowany model. Przy czym zakładamy, że metoda diagnostyczna ma dostęp do wyników działania diagnozowanego modelu na niezaburzonym, pozbawionym przykładów adwersaryjnych, zbiorze przykładów na których model ten był trenowany i walidowany.

1.4.2 Cele szczegółowe

Pierwszym celem szczegółowym jest implementacja metod atakowania modelei ML klasyfikujących dane tabelaryczne. Wybór metod ataku, wybór typów atakowanych modeli, oraz wybór zbiorów danych przetwarzanych przez te modele powinien wspierać możliwość wiarygodnego stwierdzenia skuteczności metody.

Ze względu na założenie działania metody w reżimie *czarnej skrzynki*, a co za tym idzie braku bezpośredniego dostępu do diagnozowanego modelu, drugim celem pracy jest opracowanie metody przybliżającej (aproksymującej) działanie tego modelu. Metoda aproksymująca działanie modelu poddawanego diagnostyce zapewni nam dysponowanie modelem zastępczym (ang. surrogate), którego zachowanie podlegać będzie dalszej analizie.

Najważniejszym z celów szczegółowych jest opracowanie i weryfikacja metody wykrywania ataków. Metoda opiera się na zbiorze atrybutów diagnostycznych monitorujących działanie diagnozowanego modelu oraz klasyfikatorze, który na podstawie tych atrybutów określa, czy wyniki modelu są efektem technik ataku adwersaryjnego.

Ostatnim celem szczegółowym jest weryfikacja działania opracowanej metody na nowych przykładach ataków adwersaryjnych - innych niż te, które wykorzystywano podczas tworzenia klasyfikatora wykrywającego ataki. Oprócz waloru poznawczego, cel ten ma też walor praktyczny, związany z możliwością pokrycia metodą szerokiego spectrum ataków w tym ataków jeszcze nie rozpoznanych, tak zwanych *zero day attacks*.

1.4.3 Współpraca z partnerem biznesowym

Praca została wykonana w ramach doktoratu wdrożeniowego realizowanego we współpracy z firmą QED Software sp. z o.o.[3], która specjalizuje się m.in. w rozwoju technik wyjaśnialności modeli ML. W szczególności firma bada przyczyny błędów i niepewności działania modeli oraz analizuje wpływ zmian danych wejściowych na ich funkcjonowanie. Wyniki niniejszej pracy zostaną wykorzystane w działaniach QED Software związanych z rozwojem platformy służącej do monitorowania i ciągłego audytu modeli ML.

1.4.4 Zawartość pracy

Dalsza część pracy zorganizowana jest w sposób następujący, w rozdziale drugim przedstawiono teorię i aktualny stan wiedzy dotyczące ataków adwersaryjnych na modele uczenia maszynowego. Zaprezentowano różne podejścia do klasyfikacji ataków na modele uczenia maszynowego, opierające się na bieżącym standardzie zaprezentowanym przez NIST oraz na publikacjach historycznych tworzących tę dziedzinę wiedzy. Nacisk został położony na ataki służące celowemu zaburzaniu działania

[3]https://www.qed.pl/

klasyfikatorów już po ich wytrenowaniu, gdzie klasyfikator służy przetwarzaniu danych tabelarycznych. Rozdział trzeci zawiera opis metody stanowiącej przedmiot badania w niniejszej pracy - sposobu na uzyskiwanie atrybutów diagnostycznych z monitorowanych modeli uczenia maszynowego oraz algorytmu służącego określaniu możliwego wystąpienia ataku adwersaryjnego. Kolejny rozdział przedstawia przebieg eksperymentów wraz z analizą uzyskanych wyników. Podsumowanie zawiera wnioski, umiejscowienie prac w szerszym kontekście, oraz rozważania na temat praktycznej przydatności i ograniczeń metody. W podsumowaniu przedstawiono też możliwe kierunki dalszych badań rozszerzających możliwości metody zaprezentowanej w rozprawie.

2 Ataki adwersaryjne na modele uczenia maszynowego

2.1 Podstawy teoretyczne

2.1.1 Wprowadzenie

Ataki adwersaryjne na klasyfikacyjne modele ML przeprowadzane na etapie inferencji (czyli podejmowania decyzji przez wytrenowany model) można przedstawić jako problem optymalizacji. Ich celem jest znalezienie zakłóceń (perturbacji), które prowadzą do błędnej klasyfikacji danych wejściowych. W przypadku typowych ataków adwersaryjnych rozważamy klasyfikatory, które na podstawie wektora danych wejściowych $\mathbf{x} \in \mathbb{A}$ (gdzie: $\mathbb{A}$ jest iloczynem kartezjańskim atrybutów warunkowych) przewidują klasę y za pomocą modelu $f : \mathbb{A} \to \mathbb{R}^k$, gdzie k to liczba klas, oraz gdzie $\mathcal{L}(f(\mathbf{x}'), y)$ to funkcja straty, która mierzy błąd klasyfikacji pomiędzy przewidywanymi etykietami $f(\mathbf{x}')$ a prawdziwymi etykietami y.

2.1.2 Definicja Przykładu Adwersaryjnego

Dla danej instancji wejściowej $\mathbf{x}$ i jej prawdziwej etykiety y, przykład adwersaryjny $\mathbf{x}'$ to wejście takie, że:

$$f(\mathbf{x}') \neq y, \tag{2.1}$$

przy czym $L_p(x, x') < \epsilon$, gdzie L_p to funkcja odległości, zależna od konkretnego typu ataku i jego profilu ukrywania, a ϵ jest maksymalną miarą zakłóceń, parametryzowalną w trakcie przeprowadzania ataku [3].

2.1.3 Problem Optymalizacji dla Ataków Adwersaryjnych

Matematyczny problem generowania przykładu adwersaryjnego można sformułować jako problem optymalizacji:

$$\arg\max_{\mathbf{x}'} \ \mathcal{L}(f(\mathbf{x}'), y). \tag{2.2}$$

Zakłada się przy tym, że $\mathbf{x}'$ pozostaje w określonym sąsiedztwie $\mathbf{x}$, determinowanym przez użytą funkcję odległości L_p oraz maksymalną odległość ϵ [43]. Sama optymalizacja może być przeprowadzana różnymi metodami - użyta metoda optymalizacji jest, obok użytej funkcji odległości, głównym czynnikiem rozróżniającym poszczególne typy ataków adwersaryjnych.

2.1.4 Funkcje odległości

W kontekście ataków adwersaryjnych, funkcja odległości oznacza funkcję służącą do oceny wielkości perturbacji wprowadzanej do danych wejściowych. Pomiar stopnia wprowadzonych zaburzeń jest kluczowy z punktu widzenia celu ataku - oszukania modelu przy minimalnej ingerencji w oryginalne dane. Perturbacja musi być na tyle subtelna, aby pozostać niewidoczna lub trudna do wykrycia, lecz jednocześnie wystarczająca do zmiany decyzji modelu.

Funkcja L_2, zwana również odległością Euklidesową, mierzy odległość pomiędzy oryginalnym we jściem $\mathbf{x}$ a zakłóconym wejściem $\mathbf{x}'$ jako pierwiastek z sumy kwadratów różnic między odpowiednimi cechami. Formalnie, funkcja ta definiowana jest jako:

$$\|\mathbf{x}' - \mathbf{x}\|_2 = \sqrt{\sum_{i=1}^{n} (x'_i - x_i)^2}.$$

Jest ona powszechnie stosowana w przypadku danych ciągłych w scenariuszach gdzie zakłócenia rozkładają się równomiernie po wszystkich cechach. Pozwala ona na generowanie subtelnych perturbacji trudnych do wykrycia [14, 74].

Funkcja L_∞ mierzy maksymalną wartość absolutnej różnicy między cechami oryginalnego i zakłóconego wejścia, definiowaną jako:

$$\|\mathbf{x}' - \mathbf{x}\|_\infty = \max_i |x'_i - x_i|.$$

Jest ona stosowana w metodach w których zakłada się równomierny rozkład zakłóceń, ale ograniczony do maksymalnej wartości perturbacji. Zakłócenie generowane

przy użyciu tej funkcji zazwyczaj koncentruje się na wprowadzeniu minimalnej, ale wystarczająco istotnej zmiany w newralgicznych cechach [27].

Funkcja L_0 zlicza liczbę zmienionych cech w wektorze wejściowym. Funkcja jest definiowana jako:

$$\|\mathbf{x}' - \mathbf{x}\|_0 = \sum_{i=1}^{n} \mathbb{I}(x'_i \neq x_i),$$

gdzie: $\mathbb{I}$ jest funkcją wskaźnikową, przyjmującą wartość 1, gdy elementy się różnią, oraz 0 w przeciwnym przypadku. Funkcja L_0 jest używana głównie w sytuacjach, gdzie kluczowa jest minimalizacja liczby modyfikowanych cech, np. w modelach o szczególnej wrażliwości na wybrane cechy [74].

Funkcja L_1, zwana odległością Manhattan, definiowana jako:

$$\|\mathbf{x}' - \mathbf{x}\|_1 = \sum_{i=1}^{d} |x'_i - x_i|$$

mierzy skumulowaną zmianę wszystkich cech. Jest przydatna gdy istotne jest ogólne zakłócenie w wektorze danych. Na przykład gdy interesują nas sumaryczne efekty zmian, a nie ich maksymalny lub równomierny rozkład [14].

W atakach na modele przetwarzające dane kategoryczne lub dyskretne stosuje się odległość Hamminga, która zlicza liczbę cech, w których nastąpiła zmiana wartości. Funkcję tę stosuje się w atakach na dane tabelaryczne lub tekstowe, gdzie zmienne mają ograniczoną liczbę możliwych artości [10, 32].

Powyżej wymieniono jedynie podstawowe funkcje odległości. Ogólnie, funkcje odległości pozwalają kontrolować i oceniać zakłócenia wprowadzane przez ataki adwersaryjne. Wybór odpowiedniej funkcji odległości jest kluczowy przy projektowaniu ataku, ponieważ wpływa ona na subtelność zakłóceń i skuteczność ataku. Wybór ten zależy też od specyfiki danych i celów ataku. W przypadku atrybutów kategorycznych znane metody ataku najczęściej stosują odległość Hamminga (D_H) lub L_0, natomiast w przypadku atrybutów numerycznych najczęściej stosowane są odległości L_1, L_2 oraz L_∞. W szczególności, L_2 jest często używana, gdy chcemy równomiernie rozłożyć zakłócenia. W przypadku danych opisywanych przez zarówno atrybuty numeryczne jak i kategoryczne stosuje się odpowiednio unormowane połączenia opisanych powyżej funkcji odległości [54].

2.1.5 Pomiar skuteczności ataku

Sukces ataku mierzy się wskaźnikiem zmienionych etykiet ASR (ang. Attack Success Rate) [69]:

$$\text{ASR} = \frac{\text{Liczba udanych ataków}}{\text{Całkowita liczba prób ataku}},$$

co można sformalizować jako:

$$\text{ASR} = \frac{1}{N}\sum_{i=1}^{N}\mathbb{I}(f_{\text{target}}(x_i + \delta_i) \neq y_i),$$

gdzie: N to liczba próbek, a $\mathbb{I}(\cdot)$ to funkcja indykatora.

Inną miarą stosowaną w ocenie ataków (ale również w ocenie podatności modeli na ataki) jest minimalna Wielkość Zakłócenia (Minimal Perturbation Norm). Miara ta określa najmniejszą możliwą wielkość zakłócenia δ, która wystarcza do zmiany decyzji klasyfikatora. Mniejsze wartości tej miary sugerują większą podatność modelu na subtelne zakłócenia.

Transferowalność ataków

Szczególną cechą ataków adwersaryjnych jest ich transferowalność (przenoszalność). Przykłady adwersaryjne x', zaprojektowane tak, aby oszukać model f_1, wprowadzają w różny od f_1 i niezależnie wytrenowany model f_2.

Zjawisko to wykorzystywane jest w scenariuszach, w których atakujący nie ma bezpośredniego dostępu do modelu docelowego f_{target}. W takich przypadkach wykorzystuje się model zastępczy $f_{\text{surrogate}}$ do optymalizacji zakłóceń δ tak, aby $f_{\text{surrogate}}(x+\delta) \neq y$, co często prowadzi do tego, że $f_{\text{target}}(x + \delta) \neq y$. Umożliwia to ataki typu *czarnej skrzynki*. Z drugiej strony, strategie obronne muszą uwzględniać nie tylko bezpośrednie ataki, ale także transferowane przykłady adwersaryjne, które często wykorzystują uniwersalne właściwości przestrzeni cech lub funkcji straty [78].

Czynniki wpływające na transferowalność ataków:

- Podobieństwo modeli

  Modele o podobnych architekturach lub wytrenowane na porównywalnych zestawach danych są bardziej podatne na wspólne podatności adwersaryjne. Transferowalność między modelami f_1 i f_2 można analizować na podstawie ich granic decyzyjnych B_1 i B_2, gdzie $B = \{x \mid \nabla_x \mathcal{L}(f(x), y) = 0\}$. Im większe

jest nakładanie się B_1 i B_2 w przestrzeni cech, tym wyższy jest współczynnik transferowalności $T(f_1, f_2)$ rozumiany jako:

$$T(f_1, f_2) = \mathbb{P}(f_2(x + \delta) \neq y \mid f_1(x + \delta) \neq y).$$

- Metody ataku

  Transferowalność ataku zależy od skali zaburzenia [81] oraz od użytej w ataku strategii optymalizacji. Techniki gradientowe są szczególnie skuteczne w tworzeniu ataków przenoszalnych pomiędzy modelami [76]. Przykładem jest wspomniana wcześniej metoda FGSM.

- Rozkład danych

  Modele wytrenowane na podobnych rozkładach danych $\mathcal{D}_1$ i $\mathcal{D}_2$ są podatne na podobne przykłady adwersaryjne. Niech $p(x)$ i $q(x)$ będą rozkładami danych dla modeli f_1 i f_2. Im mniejsza jest rozbieżność pomiędzy rozkładami, obliczana np. jako dywergencja Kullbacka-Leiblera $\mathrm{KL}(p\|q)$, tym większa jest transferowalność ataku.

- Właściwości funkcji straty

  Geometria przestrzeni straty (ang. loss landscape) wpływa na transferowalność przykładów adwersaryjnych. Przez przestrzeń straty rozumiemy wielowymiarową powierzchnię, o liczbie wymiarów d równej liczbie cech wejściowych używanych przez model f z parametrami θ oraz wejściem x, której każdy punkt przyjmuje wartość funkcji straty dla danej konfiguracji wartości cech wejściowych.

  $$\mathcal{L} : \mathbb{R}^d \to \mathbb{R}$$

  [49] Ataki adwersaryjnee wykorzystują obszary w przestrzeni straty gdzie niewielkie zaburzenia prowadzą do nieproporcjonalnie dużych zmian w funkcji straty - maksymalizują gradient funkcji straty $\nabla_x \mathcal{L}$ względem wejścia, co skutkuje błędnymi przewidywaniami modelu. Analizując geometrię przestrzeni straty, można wnioskować o podatności modelu na atak, oraz o transferowalności ataków między modelami.

  Model z płaskim minimum w krajobrazie funkcji straty jest bardziej odporny na ataki, ponieważ strata nie zmienia się drastycznie przy niewielkich zaburzeniach.

Z drugiej strony ostre minimum wskazuje na wysoką wrażliwość, gdzie małe zmiany mogą powodować znaczne zwiększenie straty, co czyni model bardziej podatnym na ataki adwersarialne.

Zakłócenia δ, które skutecznie wykorzystują wspólne właściwości przestrzeni straty między modelami f_1 i f_2, są bardziej transferowalne - jeśli $\nabla_x \mathcal{L}(f_1(x), y) \approx \nabla_x \mathcal{L}(f_2(x), y)$, transferowalność wzrasta.

- Odporność modeli

  Aby uodpornić modele ML na ataki adwersaryjne stosuje się rożnego rodzaju techniki. Jedną z najpopularniejszych jest trening adwersaryjny. Polega on na włączaniu do procesu treningowego specjalnie zaprojektowanych przykładów adwersaryjnych. Dzięki temu model ML uczy się rozpoznawać i ignorować takie zakłócenia, co zwiększa jego stabilność i odporność. Modele odporne f_{robust} często opierają się bezpośrednim atakom, ale mogą być nadal podatne na przykłady adwersaryjne transferowane z innych modeli. Współczynnik transferowalności ataku między modelami odpornymi, a standardowymi można definiować jako prawdopodibneństwo:

$$T(f_{\text{robust}}, f_{\text{standard}}) = \mathbb{P}(f_{\text{standard}}(x + \delta) \neq y \mid f_{\text{robust}}(x + \delta) = y).$$

2.2 Taksonomia ataków adwersaryjnych

2.2.1 Sposoby podejścia do klasyfikacji ataków adwersaryjnych

Od początku badań nad atakami na modele ML podjęto kilka prób klasyfikacji ataków adwersaryjnych. Pierwsza szersza taksonomia, zaproponowana przez Barreno i współpracowników [7], zakładała podział ataków pod kątem trzech osi: wpływu, naruszenia bezpieczeństwa oraz specyficzności. Każda z tych osi definiuje inne aspekty ataku, umożliwiając bardziej szczegółową analizę i zrozumienie sposobów, w jaki przeciwnicy mogą eksploatować systemy uczenia maszynowego.

Oś wpływu

Oś ta rozróżnia dwa główne typy ataków: kauzatywne (causative) i eksploracyjne (exploratory). Ataki kauzatywne wpływają na proces uczenia poprzez manipulację danymi treningowymi. Przykładem może być wprowadzenie do zbioru treningowego

złośliwych danych, które zmieniają sposób, w jaki model klasyfikuje przyszłe dane. Z kolei ataki eksploracyjne polegają na wykorzystaniu słabości w już wytrenowanym modelu, bez wpływania na proces jego uczenia. Przykładem jest atak polegający na wprowadzaniu danych testowych zaprojektowanych tak, aby model błędnie je klasyfikował.

Oś naruszenia bezpieczeństwa

Ataki można również klasyfikować na podstawie rodzaju naruszenia bezpieczeństwa, które powodują. Rozróżniamy tutaj ataki na integralność (integrity) oraz ataki na dostępność (availability). Ataki na integralność mają na celu spowodowanie spadku czułości klasyfikacji, w kontekście ustalonych(ej) klas(y) decyzyjnej. Przykładem może być atak, w którym złośliwe e-maile są klasyfikowane jako nieszkodliwe. Ataki na dostępność dążą do zmniejszenia specyficzności klasyfikacji czyli generowania tzw. fałszywych alarmów. Przykładem może być atak, w którym normalny ruch sieciowy jest klasyfikowany jako złośliwy, co może prowadzić do odmowy wykonania usługi (atak *denial of service*).

Oś specyficzności

Oś specyficzności odnosi się do zakresu ataku. Ataki ukierunkowane (targeted) skoncentrowane są na konkretnych instancjach danych. Przykładem jest atak, który ma na celu spowodowanie błędnej klasyfikacji określonego e-maila lub wiadomości. Ataki nieselektywne (indiscriminate) obejmują szeroką klasę danych wejściowych. Przykładem może być atak polegający na wprowadzeniu dużej liczby zróżnicowanych złośliwych danych, które powodują ogólną degradację wydajności modelu.

Powyższa klasyfikacja zakłada określony model zagrożeń dla ataków adwersaryjnych, w którym zakłada się dwa główne obszary analizy: profil atakującego i cele bezpieczeństwa.

Profil atakującego

Model zagrożeń definiuje profil atakującego, opisując jego motywacje i możliwości:

- Motywacje atakującego: Atakujący może dążyć do uzyskania dostępu do zasobów systemu (poprzez fałszywe decyzje modelu ML) lub do zakłócenia normalnych operacji systemu (poprzez generowanie dużej liczby fałszywych

alarmów). Przykładowo, autor wirusa chce, aby wirus przeszedł przez filtr i zainfekował system, podczas gdy nieuczciwy przedsiębiorca może chcieć zablokować ruch do sklepu internetowego konkurenta.

- Możliwości atakującego: Zakłada się, że atakujący ma wiedzę na temat algorytmu uczącego się i w wielu przypadkach częściową lub pełną wiedzę na temat zbioru treningowego, na przykład może podsłuchiwać ruch sieciowy w czasie zbierania danych treningowych przez uczący się system. Atakujący może być zdolny do generowania lub modyfikowania danych używanych do treningu; w niektórych przypadkach atakujący może kontrolować część danych treningowych. Ogólnie zakłada się, że atakujący może generować dowolne instancje danych, choć w praktyce mogą istnieć ograniczenia związane z kontekstem ataku, takie jak brak możliwości zmiany etykiet danych w zbiorze treningowym.

Cele bezpieczeństwa

W kontekście uczenia maszynowego cele bezpieczeństwa systemów obejmują dwa główne aspekty związane z wspominaną już integralnością i dostępnością. W praktyce, w zależności od konkretnego przeznaczenia systemu korzystającego z modułu ML integralność i dostępność interpretowana może być różnie. Przykładowo, w systemach związanych z cyberbezpieczeństwem integralność oznacza, że atakujący nie mogą uzyskać dostępu do zasobów systemu przez błędne zaklasyfikowanie złośliwych danych jako bezpiecznych. Naruszenie tego celu oznacza, że złośliwe dane mogą przejść przez system bez wykrycia. Dostępność oznacza zaś, że normalne operacje systemu nie będą zakłócane przez błędne zaklasyfikowanie bezpiecznych danych jako złośliwych. Naruszenie tego celu prowadzi do fałszywych pozytywów, co może skutkować odmową dostępu do zasobów przez legalnych użytkowników.

Przedstawiony powyżej model zagrożeń nie dotyka takich zagadnień jak skala ataku. Atakujący może mieć pod tym względem różny cel. Może mu zależeń na całkowitym zaburzeniu pracy systemu opartego o model uczenia maszynowego, lub też może chcieć jedynie obniżyć jakość jego działania (np w przypadku celu będącego erozją zaufania do działań systemu).

2.2.2 Cykl życia ataków

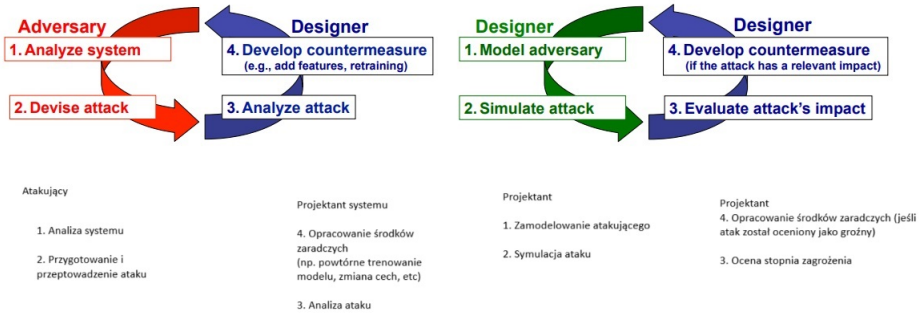
Cykl uczenia maszynowego narażonego na ataki adwersaryjne (Adversarial Machine Learning Cycle - AMLC) jest modelem dynamicznej interakcji pomiędzy systemami uczenia maszynowego a adwersarzami. Proces ten można postrzegać jako grę, w której algorytmy uczą się z istniejących danych, aby prawidłowo klasyfikować lub przewidywać nowe dane, podczas gdy atakujący, bazując na obserwowanym stanie systemów, starają się zmieniać dane wejściowe lub parametry algorytmu, aby wywołać błędną klasyfikację lub błędne przewidywania. W odpowiedzi na ataki, systemy zawierające modułu ML mogą stosować różne mechanizmy obronne:

- wykrywanie nietypowych zjawisk i anomalii w pracy systemów;
- dokonywać detekcji ataków;
- wykrywać typy ataków;
- modyfikować działanie systemu w bezpośredniej odpowiedzi na wykryty atak o określonym typie;
- stosować metody uodparniania modeli ML na ataki.

Ten cykl wzajemnych działań między adwersarzami a obrońcami, gdzie każda ze stron stara się przechytrzyć przeciwnika, oznaczany jest jako AMLC. AMLC stanowi formę gry pomiędzy atakującym modele ML, a broniącym i polega na ciągłej rywalizacji, w której każda ze stron stara się opracować bardziej zaawansowane techniki, aby zyskać przewagę nad przeciwnikiem. Proaktywne podejście do bezpieczeństwa nakazuje by twórca systemu wcielił się w rolę atakującego, i analizował tworzony system również pod kątem jego podatności na ataki [9].

2.2.3 Klasyfikacja ataków adwersaryjnych według NIST

National Institute of Standards and Technology (NIST) skodyfikował i opublikował w styczniu 2024 wszechstronną taksonomię ataków adwersaryjnych na modele uczenia maszynowego, uwzględniającą kluczowe wymiary, takie jak etap cyklu życia modelu, cel i motywacje atakujących, zakres wiedzy i możliwości atakującego oraz modalność danych [57]. W związku z rolą pełnioną przez NIST w świecie inżynierii, to podejście z miejsca stało się standardem w systematyzacji ataków i dyskusjach o związanych z nimi zagrożeniach i strategiach obronnych. Dla ataków na modele ML przedstawia się ono następująco.



Rys. 2.1: Schematyczna reprezentacja rywalizacji pomiędzy atakującym, a projektantem systemu (wariant reaktywny i proaktywny) [9].

Etap cyklu życia modelu

Kluczowym aspektem taksonomii NIST jest etap cyklu życia modelu, na którym przeprowadzany jest atak. Wyróżnia się dwa główne etapy: etap treningu oraz etap inferencji. Ataki na etapie treningu mają na celu zakłócenie procesu uczenia, na przykład poprzez wprowadzenie złośliwych danych do zbioru treningowego, co prowadzi do niepożądanych efektów w finalnym modelu. W atakach na etapie inferencji atakujący manipuluje danymi wejściowymi dostarczanymi do modelu już wytrenowanego, aby uzyskać błędne predykcje. Oba typy ataków mogą wpływać na funkcjonalność modeli w czasie rzeczywistym, mając znaczący wpływ na integralność decyzji podejmowanych przez systemy oparte na modelach ML.

Cel i motywacja atakującego

Kolejnym istotnym wymiarem klasyfikacji jest cel i motywacja atakującego. NIST wskazuje na trzy główne kategorie: ataki na integralność, dostępność oraz prywatność modelu. Ataki na integralność mają na celu wprowadzenie błędów w decyzjach podejmowanych przez model. Ataki na dostępność dążą do zakłócenia lub zablokowania działania modelu, co może mieć poważne konsekwencje - zwłaszcza w systemach krytycznych dla bezpieczeństwa. Z kolei ataki na prywatność koncentrują się na uzyskiwaniu informacji o danych treningowych, cechach użytkowników lub samych parametrach modelu, co stanowi zagrożenie dla poufności informacji przechowywanych przez system.

Wiedza i możliwości atakującego

Zakres wiedzy i możliwości atakującego jest kolejnym kryterium, które różnicuje metody ataków adwersaryjnych. Możemy wyróżnić ataki w trybie *białej skrzynki*, *czarnej skrzynki* oraz *szarej skrzynki*. W trybie *białej skrzynki* atakujący ma pełny dostęp do modelu, jego struktury i parametrów, co pozwala na precyzyjną manipulację danymi wejściowymi. Ataki typu *czarnej skrzynki* zakładają, że model jest nieznany, a atakujący bazuje jedynie na wynikach uzyskanych na podstawie zapytań do modelu. Tryb *szarej skrzynki* obejmuje sytuacje pośrednie, w których atakujący ma dostęp jedynie do częściowych informacji, takich jak architektura modelu, lecz bez znajomości wag i gradientów. Każdy z tych scenariuszy wymaga innych technik, które mogą być bardziej lub mniej skuteczne w zależności od poziomu dostępnych informacji.

Modalność danych

Ostatnim głównym wymiarem taksonomii NIST jest modalność danych, czyli rodzaj i struktura danych wejściowych, na których przeprowadzane są ataki. Modalność ta określa specyficzne podejścia ataków dostosowane do formatu danych, takich jak obrazy, tekst czy dźwięk. W przypadku obrazów ataki mogą obejmować subtelne manipulacje pikselami, które nie są zauważalne dla ludzkiego oka, ale skutecznie zakłócają predykcję modelu. W przypadku tekstu ataki mogą polegać na modyfikacji słów lub parafrazowaniu, co wpływa na interpretację treści przez modele przetwarzania języka naturalnego. Z kolei w przypadku danych akustycznych mogą być stosowane zmiany w częstotliwościach lub wstawianie zakłócających dźwięków, aby wprowadzić błąd w rozpoznawaniu mowy. Każda z tych modalności wymaga dostosowania strategii ataku do właściwości danych. Jako odrębne modalności, generujące specyficzne wyzwania, definiuje się dane tabelaryczne oraz dane cyberbezpieczeństwa (połączenie szeregów czasowych, danych tabelarycznych oraz danych tekstowych).

Inne istotne kryteria

Oprócz głównych wymiarów, NIST wskazuje również na dodatkowe kryteria, które mogą być istotne w analizie zagrożeń związanych z atakami adwersaryjnymi. Przykładowo, efektywność zapytań jest kluczowym aspektem, zwłaszcza w atakach typu *czarnej skrzynki*, gdzie liczba zapytań może być ograniczona przez zasoby lub mechanizmy obronne. Kolejnym kryterium jest odporność na detekcję, czyli zdolność ataku do pozostawania niezauważonym przez systemy wykrywające anomalie.

2.2.4 Umiejscowienie niniejszej pracy

Odnosząc się do taksonomii NIST, której podsumowanie przedstawione jest w tabeli 2.2, niniejsza praca skupia się na wykrywniu ataków na integralność modelu, dokonywanych na etapie inferencji z wykorzystaniem wcześniej wytrenowanego modelu, należących do typu ataków omijania (ang. evasion attacks). Dodatkowo zakłada się, że metoda wykrywania ataków opracowana w ramach realizacji doktoratu bazuje na założeniach *czarnej skrzynki*. Oznacza to, że zakłada się, że metoda ma dostęp jedynie do inferencji modelu, do oryginalnego zbioru przykładów, na którym trenowana i walidowana była wersja prawidłowo działającego (tj. niezaatakowanego) modelu ML będącego przedmiotem diagnostyki.

2.3 Przegląd metod ataków

Poniżej zaprezentowano krótki przegląd głównych metod ataków na modele ML służące klasyfikacji danych wejściowych - ataków dokonywanych na etapie inferencji modelu.

2.3.1 Ataki z dostępem do atakowanego modelu

W scenariuszach *białej skrzynki* atakujący posiada pełny dostęp do modelu, co oznacza, że zna architekturę i wartości parametrów modelu. W scenariuszu tym atakujący może również śledzić - dla zadanej grupy modeli np. sieci neuronowych - wartości wpływające na proces wypracowywania decyzji modelu. Dzięki temu możliwe jest precyzyjne kierowanie perturbacjami w celu uzyskania maksymalnej skuteczności przy minimalnych zmianach w danych wejściowych. W literaturze wyróżnia się kilka głównych podejść w tej kategorii.

Ataki jednokrokowe

W literaturze przedmiotu ataki jednokrokowe znane są jako Signgle-Step-Attacks. Przykładem metody jednokrokowej jest Fast Gradient Sign Method (FGSM), zaproponowana przez Goodfellowa i in. [27]. FGSM zakłada generowanie perturbacji w jednym kroku poprzez dodanie niewielkiej modyfikacji w kierunku znaku gradientu funkcji straty względem danych wejściowych. Perturbacja jest wyrażona jako:

$$x' = x + \epsilon \cdot \text{sign}(\nabla_x \mathcal{L}(f(x), y))$$

Klasyfikacja Ataków	Opis	Rodzaje Ataków
Etap cyklu uczenia	Wskazuje fazę cyklu życia modelu uczenia maszynowego, w której dochodzi do ataku.	- **Ataki na zbieranie danych**: Modyfikowanie danych na etapie ich zbierania. - **Ataki na etapie treningu**: Dodawanie złośliwych danych do zbioru treningowego. - **Ataki na etapie wnioskowania**: Tworzenie wejść, które oszukują model podczas procesu przewidywania.
Cele atakującego	Określa, jakie cele chce osiągnąć atakujący.	- **Ataki na integralność**: Powodowanie, że model generuje błędne przewidywania. - **Ataki na dostępność**: Zakłócenie działania modelu, aby przestał działać poprawnie. - **Ataki na prywatność**: Wydobywanie poufnych informacji z modelu, takich jak dane treningowe.
Możliwości atakującego	Opisuje poziom wiedzy i zasoby, jakimi dysponuje atakujący.	- **Ataki typu white-box (biała skrzynka)**: Atakujący ma pełny dostęp do architektury modelu, parametrów i wag. - **Ataki typu black-box (czarna skrzynka)**: Atakujący nie ma wiedzy o działaniu wewnętrznym modelu, ale może wysyłać zapytania do modelu i obserwować odpowiedzi.
Konkretne rodzaje ataków	Grupa konkretnych technik atakowania modeli uczenia maszynowego.	- **Ataki omijające (evasion attacks)**: Modyfikowanie danych wejściowych, aby model błędnie je zaklasyfikował (np. zmiana kilku pikseli w obrazie). - **Ataki zatruwające (poisoning attacks)**: Wprowadzanie złośliwych danych do zbioru treningowego, aby zmienić zachowanie modelu. - **Ataki odtwarzania danych (model inversion attacks)**: Odtwarzanie oryginalnych danych wejściowych na podstawie wyników modelu. - **Ataki ujawniające członkostwo (membership inference attacks)**: Ustalanie, czy dany element danych (np. konkretne zdjęcie) został użyty w zbiorze treningowym modelu.

Rys. 2.2: Podsumowanie klasyfikacji ataków według NIST.

gdzie: ϵ jest parametrem kontrolującym siłę perturbacji, $f(x)$ to model, $\mathcal{L}$ to funkcja straty, a $\nabla_x \mathcal{L}$ oznacza gradient funkcji straty względem wejścia x. Jest to jedna z szybszych i prostszych w użyciu metod ataków.

Ataki Wielokrokowe

Bardziej zaawansowane są metody ataków wielokrokowych znanych równiez jako ataki iteracyjne (Iterative Attacks). W tego typu atakach perturbacja generowana jest iteracyjnie, co umożliwia bardziej precyzyjne dostosowanie modyfikacji do specyfiki modelu. Przykładem takiego ataku jest Projected Gradient Descent (PGD) [52]. Atak ten szczególnie sprawdza się w scenariuszach, gdzie celem jest uzyskanie adwersaryjnych przykładów odpornych na mechanizmy obronne. Podstawą PGD jest transformacja:

$$x'_{t+1} = \text{Proj}_\epsilon \left(x'_t + \alpha \cdot \text{sign}(\nabla_x \mathcal{L}(f(x'_t), y)) \right)$$

gdzie: Proj_ϵ to projekcja na kulę o promieniu ϵ, a α oznacza krok iteracji. Dzięki tym iteracjom PGD jest bardziej skuteczny od FGSM, jednak wymaga większej liczby obliczeń.

Ataki maksymalizujące zakłócenie

Jednym z bardziej zaawansowanych ataków, który koncentruje się na maksymalizowaniu błędów modelu przy jednoczesnej minimalizacji zauważalności perturbacji, jest metoda opracowana przez Carliniego i Wagnera [14], należąca do grupy ataków maksymalizujących zakłócenie (Maximally Disruptive Attacks). W metodzie tej optymalizacja ataku odbywa się poprzez minimalizację odległości między oryginalnym, a adwersaryjnym przykładem przy jednoczesnym zapewnieniu błędu klasyfikacji:

$$\min_{x'} \|x' - x\|_p + c \cdot \mathcal{L}(f(x'), y')$$

gdzie: y' jest docelową etykietą klasyfikacyjną, a c to parametr pozwalający na dostosowanie siły perturbacji do celu klasyfikacyjnego. Atak ten wymaga złożonej obliczeniowo optymalizacji, lecz jest bardzo skuteczny w omijaniu zaawansowanych mechanizmów obronnych.

Ataki na bazie gradientu drugiego rzędu

Kolejna grupa metoda otrzymywania precyzyjnych perturbacji to ataki wykorzystujące gradienty drugiego rzędu (Second-Order Gradient Attacks). Przykładowo,

metoda przedstawiona w pracy [56] wykorzystuje gradienty (pochodne) drugiego rzędu do bardziej precyzyjnego zbliżenia się do granicy decyzyjnej modelu. Każdy krok iteracyjnie przybliża punkt do najbliższej granicy decyzyjnej:

$$x'_{t+1} = x_t + \frac{-\mathcal{L}(f(x_t), y)}{\|\nabla_x \mathcal{L}(f(x_t), y)\|_2^2} \nabla_x \mathcal{L}(f(x_t), y)$$

Ataki odporne na przekształcenia danych

W rzeczywistych zastosowaniach aplikacjach często konieczne jest wykonywanie pewnych operacji (np. skalowania, rotacji) na danych podlegających klasyfikacji. Wymienione wcześniej metody generowania przykładów adwersaryjnych nie są odporne na takie przekształcenia, oznacza to, że przekształcone przykłady często tracą swoją adwersaryjną naturę. Ataki odporne na przekształcenia danych znane są jako ataki typu *Expectation over Transformation* i mają one głownie zastosowanie do metod klasyfikacji obrazów [5]. Metody te dążą do osiągnięcia odporności na przekształcenia danych, minimalizując funkcję straty oczekiwaną względem losowych przekształceń:

$$x' = \arg\min_{x'} \mathbb{E}_{t\sim T} \left[\mathcal{L}(f(t(x')), y)\right]$$

gdzie: T to zbiór przekształceń, a $\mathcal{L}$ to funkcja straty.

2.3.2 Ataki bez dostępu do modelu atakowanego

W scenariuszu *czarnej skrzynki* atakujący nie ma dostępu do modelu ani jego parametrów. Z tego powodu niezbędne są bardziej wyrafinowane techniki optymalizacyjne umożliwiające skuteczne omijanie klasyfikatorów przy ograniczonej wiedzy o modelu [25, 38]. Metody te zawierają elementy estymacji gradientu, estymacji granicy decyzyjnej, lub odtworzenia atakowanego modelu [83].
Ataki typu black-box można sklasyfikować na podstawie informacji, jakie atakujący może uzyskać z modelu docelowego f, który mapuje dane wejściowe

$$x \in \mathcal{X} \quad \text{na wyjścia} \quad y \in \mathcal{Y}.$$

- Dostępność tylko etykiet (ang. label-only attacks)

  W scenariuszu tym model f_{target} traktowany jest jako czarna skrzynka – dostępne są jedynie dyskretne wyniki klasyfikacji - etykiety - dla danych wejściowych x:

  $$f_{\text{target}}(x) \rightarrow \{y, p(y)\}.$$

Nie są dostępne prawdopodobieństwa ani gradienty. Perturbacje muszą być generowane na podstawie heurystyk lub eksploracji przestrzeni wejściowej. Kluczowym celem jest zlokalizowanie granic decyzji modelu i minimalizacja odległości między oryginalnym punktem x a granicą $\partial f(x)$, przy zachowaniu błędnej klasyfikacji. Większość ataków label-only stara się generować minimalne perturbacje δ, aby były trudniejsze do wykrycia.

- Dostępność tylko decyzji modelu (ang. decision-only attacks)

  Ataki te (zwane też "hard-label attacks") to klasa ataków adwersaryjnych typu *czarnej skrzynki*, w której atakujący ma dostęp wyłącznie do informacji o poprawności lub błędności klasyfikacji modelu docelowego. Atakujący wie, czy wejście x zostało poprawnie sklasyfikowane ($f(x) = y$), ale nie zna zwróconej etykiety ani prawdopodobieństw klas. Brak jest dostępu do etykiet klas, logitów lub prawdopodobieństw. Ze względu na brak dostępu do szczegółowych wyników, gradienty nie mogą być bezpośrednio obliczone ani przybliżone, co wymusza stosowanie metod heurystycznych lub iteracyjnych eksploracji przestrzeni wejściowe w celu zlokalizowania granic decyzji modelu $\partial f(x)$ w przestrzeni wejściowej X, aby znaleźć minimalną perturbację δ, która powoduje zmianę decyzji modelu.

- Dostępność prawdopodobieństwa (ang. score-based attacks)

  W tym scenariuszu atakujący ma dostęp nie tylko do zwracanych przez system etykiet, ale też do prawdopodobieństwa klas - wektora $f(x) = [p_1, p_2, \ldots, p_k]$, gdzie p_i to prawdopodobieństwo przypisania wejścia x do klasy i, lub też do wartości logitów - surowych wyników przed ich przekształceniem w prawdopodobieństwa przez funkcję aktywacji. Dzięki dostępowi do wyników modelu, atakujący może przybliżać gradienty $\nabla_x \mathcal{L}(f(x), y)$, co pozwala stosować dokładniejsze techniki eksplorowania przestrzeni wejściowej niż w pozostałych scenariuszach, oraz stosować techniki optymalizacji zbliżone do tych z ataków z pełnym dostępem do modeli.

Przy takim założeniu, w zależności od podejścia, zakłócenia δ wprowadzane są za pomocą:

- Metod bez gradientu

  Metody te są stosowane, gdy gradienty modelu lub jego przybliżenia nie są bezpośrednio dostępne. Opierają się na próbkowaniu przestrzeni wejściowej

i iteracyjnej optymalizacji. W podejściach opartych o losowe wyszukiwanie (ang. Random Search), następuje losowe generowanie perturbacji δ i wybór tej, która maksymalizuje stratę lub zmienia decyzję modelu. Algorytm:

- Wybierz k losowych perturbacji $\{\delta_1, \delta_2, \ldots, \delta_k\}$.
- Znajdź δ^*, które zmienia klasyfikację:

$$\delta^* = \arg\min_{\delta_i} \|\delta_i\|, \quad \text{przy } f(x + \delta_i) \neq f(x).$$

Takie podejście może się sprawdzać w niskowymiarowych przestrzeniach wejściowych. Dla przestrzeni wysokowymiarowych stosuje się podejście SPSA (ang. Simultaneous Perturbation Stochastic Approximation), w którym przybliża się gradienty poprzez losowe perturbacje w przestrzeni wejściowej.

$$\hat{\nabla}_x \mathcal{L} \approx \frac{\mathcal{L}(x + \delta, y) - \mathcal{L}(x - \delta, y)}{2\|\delta\|} \cdot \delta,$$

gdzie δ to losowy wektor [75].

- Metod próbkowania granicy decyzyjnej

  Tak klasa metod opiera się na optymalizacji zaburzenia przez eksplorację granic decyzyjnych modelu, zwykle zaczynając od perturbacji położonych daleko od tych granic - rozpoczynając od x', które jest już błędnie klasyfikowane i iteracyjnie minimalizując zaburzenie $\|\delta\|$. W najprostszym ataku typu Boundary Attack [13] aktualizacja zaburzenia odbywa się poprzez:

$$x'_{t+1} = x'_t + \alpha \cdot \frac{x - x'_t}{\|x - x'_t\|},$$

  gdzie α to krok optymalizacji. Rozwój tych metod skupia się na szukaniu metod redukcji liczby zapytań do modelu, przybliżających gradienty wzdłuż granicy decyzji. Przykładem takiego wydajnego ataku jest atak HopSkipJump. W metodzie tej, kolejne przybliżenie perturbacji uzyskuje się z użyciem przekształcenia:

$$x'_{t+1} = x'_t + \alpha \frac{x - x'_t}{\|x - x'_t\|_2} + \beta v$$

  gdzie: α to krok zbliżania do x, a β kontroluje losowy ruch [17].

- Metod gradientowych

Metody te zakładają możliwość przybliżenia gradientu funkcji straty - lub wręcz stworzenia modelu zastępczego dla modelu atakowanego - a następnie przeprowadzenie ataków bazujących na tej wiedzy.

Do tej klasy metod należą też metody przybliżające (odtwarzające) gradient na podstawie wyników zwracanych przez model, w sytuacji gdy dostępne są prawdopodobieństwa klas lub logity. Możliwa jest wtedy estymacja gradientów na podstawie różniczkowania skończonego.

$$\frac{\partial \mathcal{L}}{\partial x_i} \approx \frac{\mathcal{L}(x + \epsilon e_i, y) - \mathcal{L}(x, y)}{\epsilon},$$

gdzie e_i to jednostkowy wektor. W metodzie ZOO [18] wprowadza się kierunkowe zaburzenia,

$$x' = x + \eta \cdot \text{sign}(\nabla_x \mathcal{L}).$$

do momentu osiągnięcia maksymalnej liczby iteracji lub osiągnięcia zmiany decyzji.

W przypadku gdy istnieje możliwość stworzenia modelu zastępczego f_{proxy}, który przybliża f, perturbacje mogą być generowane na f_{proxy}, a następnie przenoszone na f dzięki wykorzystaniu cechy transferowalności ataków [61]. Można w takim scenariuszy skorzystać z szerokiego wachlarza dostepnych metod typu *białej skrzynki*. Przykład tego podejścia to FGSM na modelu zastępczym f_s:

$$x' = x + \epsilon \cdot \text{sign}(\nabla_x \mathcal{L}(f_s(x), y))$$

- Metod heurestycznych

  Zestaw metod opierających się na regułach empirycznych, intuicji lub ograniczonej eksploracji przestrzeni wejściowej, aby generować skuteczne perturbacje δ. W przeciwieństwie do metod gradientowych czy optymalizacyjnych, heurystyki zazwyczaj nie wykorzystują gradientów ani precyzyjnej funkcji celu. Zamiast tego eksplorują przestrzeń wejściową w sposób zgodny z ograniczeniami domeny danych lub bazują na prostych strategiach maksymalizacji efektu zaburzeń. Są szczególnie przydatne w przypadku danych tabelarycznych, tekstowych lub innych strukturach, gdzie gradienty mogą być trudne do zastosowania lub interpretacji. Przykładowo, dla ataków na dane tekstowe, możliwą heurestyką jest zamiana słów na synonimy lub losowe manipulacje znakami i tokenami, aby

zmienić klasyfikację [1, 21]. Dla danych tabelarycznych, przykładowym standardowym podejściem jest permutacja wartości cech, z zachowaniem ograniczeń domenowych oraz wyborze najmniejszej skutecznej permutacji [32]. Samo generowanie zaburzeń morze odbywać się w sposób losowy lub z wykorzystaniem algorytmów ewolucyjnych. Przykładowo, w metodzie GenAttack [2], przestrzeń wejściowa eksplorowana jest w celu maksymalizacji funkcji straty $\mathcal{L}(f(x+\delta), y)$ z użyciem algorytmu:

- Inicjalizacja
  - Utwórz początkową populację perturbacji $\delta_1, \delta_2, \ldots, \delta_N$, gdzie N to rozmiar populacji.
  - Początkowe perturbacje są losowe, ograniczone parametrem $\|\delta\|_p \leqslant \epsilon$.
- Ocena:
  - Oblicz wartość funkcji straty $\mathcal{L}(f(x+\delta_i), y)$ dla każdej perturbacji w populacji.
  - Wybierz najlepsze perturbacje jako kandydatów do następnych kroków.
- Operatory genetyczne:
  - Krzyżowanie (ang. crossover): Połącz dwie perturbacje, aby wygenerować nowe perturbacje.
  - Mutacja: Wprowadź losowe zmiany w perturbacjach, aby eksplorować przestrzeń wejściową.
  - Selekcja: Wybierz najlepsze perturbacje na podstawie wartości funkcji straty.
- Aktualizacja populacji: zastąp starą populację nową generacją perturbacji.
- Warunek stopu - zatrzymaj optymalizację, gdy osiągnięto maksymalną liczbę iteracji lub gdy perturbacja δ spełnia warunki ataku (np. $f(x+\delta) \neq y$).

Takie podejście może być stosowany w różnych domenach danych, przy minimalnych założeniach dotyczących struktury modelu.

W przypadku ataków *czarnej skrzynki* wykorzystywanych praktycznie, pojawia się też dodatkowe zagadnienie – ograniczenia liczby zapytań do modelu atakowanego w celu ukrycia samego faktu zwiększonego obciążenia tego modelu [80].

2.4 Metody wykrywania ataków

Wykrywanie ataków adwersaryjnych sprowadza się do stwierdzenia kiedy dane wejściowe stanowią celowo zmodyfikowane przykłady adwersaryjne.

Detekcja przykładów adwersaryjnych to zagadnienie, w którym oceniamy czy dane wejściowe $\mathbf{x}'$ są potencjalnym przykładem adwersaryjnym. Wykorzystuje się tutaj odległość wejścia $\mathbf{x}'$ od rozkładu danych uczących lub granicy decyzyjnej modelu. Formalnie, jak kryterium wykrywania można przyjąć spełnienie warunku odległości:

$$L(\mathbf{x}', \mathcal{D}) > \delta,$$

gdzie $L(\mathbf{x}', \mathcal{D})$ to funkcja odległości $\mathbf{x}'$ od rozkładu uczącego $\mathcal{D}$, a δ to próg detekcji.

Wykrywanie ataków adwersaryjnych - metody *białej skrzynki*

Metody *białej skrzynki* zakładają pełny dostęp do modelu, w tym do jego parametrów, struktur warstw i gradientów. Dzięki temu możliwe jest analizowanie wewnętrznych reprezentacji modelu.

Metody bazujące na analizie wartości aktywacji modelu.
Metody te wykorzystują dostęp do wewnętrznych warstw modelu i analizują wartości aktywacji neuronów w odpowiedzi na dane wejściowe. Przykładem jest użycie odległości Mahalanobisa do analizy wektorów aktywacji w określonych warstwach modelu. Dla próbki x, odległość między jej wektorem aktywacji $a^l(x)$, a średnim wektorem aktywacji μ^l w warstwie l można wyrazić jako:

$$L_M(x) = \sqrt{(a^l(x) - \mu^l)^T (\Sigma^l)^{-1} (a^l(x) - \mu^l)},$$

gdzie Σ^l to macierz kowariancji dla warstwy l. Wysokie wartości $L_M(x)$ mogą wskazywać na dane celowo zaburzone. [51] [28].

Metody bazujące na spójności predykcji i czułości modelu
Przykłady adwersaryjne są przygotowywane w celu wykorzystywania luk w modelu, co prowadzi do charakterystycznych zachowań, takich jak niska pewność predykcji lub nadmierna czułość na perturbacje. Umożliwia to projektowanie metod wykrywania ataków bazujących na analizie wartości pewności w predykcji (softmax confidence):

$$S(x) = \max_j p_\theta(y = j \mid x).$$

Próbki z $S(x) < \tau$ są oznaczane jako podejrzane [35]. Przykłady adwersaryjne są wrażliwe na drobne perturbacje. Można to wykorzystać w wykrywaniu ataków

poprzez stworzenie równoległego przepływu danych, w których dane wejściowe są celowo zaburzane i oceniana jest stabilności predykcji wobec małych zakłóceń δ. Poszukuje się niespójności:

$$\Delta(f_\theta, x) = ||f_\theta(x) - f_\theta(x + \delta)||_p.$$

Metody wykorzystujące właściwości gradientowe modelu.
Metody te analizują gradienty funkcji straty względem danych wejściowych. Pełen dostęp do modelu umożliwia obliczenie gradientów $\nabla_x \mathcal{L}(f(x), y)$, gdzie $\mathcal{L}$ to funkcja straty, $f(x)$ to predykcja modelu, a y to etykieta prawdziwa. Nietypowe wartości gradientów mogą sygnalizować ataki. [30] [59].

Wykrywanie ataków adwersaryjnych - metody *czarnej skrzynki*

Metody *czarnej skrzynki* działają bez potrzeby pełnego dostępu do struktury modelu lub jego parametrów. Opierają się na analizie wyników modelu lub cech danych wejściowych. Są przez to bardziej uniwersalne, ale też mniej precyzyjne niż podejścia zakładające pełen dostęp do modelu.

Metody bazujące na analizie statystycznej danych wejściowych.
Te metody opierają się na analizie rozkładów cech danych wejściowych, badając, czy istnieją znaczące różnice między rozkładami próbek naturalnych i adwersaryjnych. Przykłady adwersaryjne znajdują się często poza rozkładem danych niezaburzonych [29]. Do porównania rozkładów można użyć np. dywergencji Kullbacka-Leiblera:

$$D_{\mathrm{KL}}(P_X \parallel P_{X_{adv}}) = \sum_x P_X(x) \log \frac{P_X(x)}{P_{X_{adv}}(x)},$$

gdzie: P_X i $P_{X_{adv}}$ to rozkłady cech odpowiednio dla danych naturalnych i adwersaryjnych. Wysokie wartości D_{KL} sugerują różnicę między próbkami naturalnymi a adwersaryjnymi [26].

Do tej kategorii metod należą metody bazujące na wykrywaniu anomalii w przestrzeni danych wejściowych.
Ta grupa metod bazuje na znanych podejściach do identyfikacji anomalii w danych. Metody wykrywania anomalii polegają na identyfikowaniu próbek, które różnią się od oczekiwanego rozkładu danych D_{clean}. W przypadku ataków adwersaryjnych, próbki x_{adv} celowo naruszają struktury i właściwości danych. Ataki adwersaryjne mogą powodować przesunięcie próbek w przestrzeni cech $\phi(x)$ [79]. Można je wykrywać poprzez pomiar odległość od klastrów reprezentujących dane czyste:

$$d(x) = ||\phi(x) - \mu_{\mathrm{clean}}||_2,$$

gdzie μ_{clean} to centroid klastra danych czystych. Jako metodę detekcji anomalii często wykorzystuje się lasy izolacyjne [50] [24].

Metody bazujące na autoenkoderach.
Autoenkodery są kolejnym narzędziem nadającym się do wykrywania ataków adwersaryjnych. Autoenkoder składa się z enkodera f i dekodera g, które uczą się minimalizować błąd rekonstrukcji:

$$\min_{\theta_f,\theta_g} \|x - g(f(x;\theta_f);\theta_g)\|,$$

gdzie: θ_f i θ_g to odpowiednio parametry enkodera i dekodera. Autoenkodery trenowane na danych niezaburzonych mają trudności z rekonstrukcją danych adwersaryjnych [63], co skutkuje wysokim błędem rekonstrukcji:

$$\mathcal{L}_{\text{recon}} = ||x - f_{\text{auto}}(x)||_p.$$

Pozwala na ich identyfikację danych zaburzonych [68] [84].

Metody bazujące na ocenie zaufania modelu.
Metody te oceniają poziom zaufania modelu względem jego predykcji. Metody wymagają jedynie dostępu do decyzji modelu. Wysoka entropia predykcji:

$$H(x) = -\sum_{i=1}^{C} p(y_i|x) \log p(y_i|x),$$

gdzie: $p(y_i|x)$ to prawdopodobieństwo przypisania klasy i próbce x może wskazywać na próbki adwersaryjne, ponieważ model jest mniej pewny swoich predykcji [35].

2.5 Specyfika ataków adwersaryjnych na modele przetwarzające dane tabelaryczne

Ataki adwersaryjne na modele przetwarzające dane tabelaryczne różnią się od tych stosowanych w analizie obrazów czy tekstu [46]. Dane tabelaryczne charakteryzują się różnorodnością typów cech, ograniczeniami domenowymi oraz złożonymi zależnościami między atrybutami opisującymi przykłady. Specyfika ta wymaga dostosowania zarówno strategii ataków, jak i metod ich wykrywania.

2.5.1 Różnorodność typów cech

Dane tabelaryczne zawierają różne typy zmiennych wymagających odmiennego podejścia w kontekście ataków adwersaryjnych. Zmienne numeryczne wymagają

precyzyjnych modyfikacji, aby zachować ich realistyczny charakter w granicach dopuszczalnych wartości, np. wiek lub wynagrodzenie. Zmienne kategoryczne, takie jak zawód czy lokalizacja, mogą być zmieniane jedynie na inne wartości należące do ustalonego zbioru kategorii. Szczególnym przypadkiem są zmienne binarne, które mimo prostej struktury mogą być kluczowe dla klasyfikacji. Modyfikacja zmiennej binarnej będzie zawsze maksymalna, a przez to łatwa w wykryciu. [32].

2.5.2 Ograniczenia domenowe

Dane tabelaryczne często podlegają ścisłym ograniczeniom, wynikającym z reguł biznesowych lub rzeczywistych relacji między cechami. W zestawach danych medycznych wskaźniki takie jak BMI muszą być zgodne z wartościami wzrostu i wagi, a w danych finansowych zmienne jak wiek i dochód muszą mieścić się w realistycznych przedziałach. Istotne są również ograniczenia czasowe, gdzie daty muszą zachowywać logiczną sekwencję (np. data rozpoczęcia wcześniejsza niż zakończenia) i nie mogą wskazywać na przyszłość w przypadku danych historycznych. Ataki muszą uwzględniać te ograniczenia, aby generowane przykłady adwersaryjne nie były łatwo wykrywalne jako anomalie [8, 16].

2.5.3 Zachowanie spójności między wartościami atrybutów

W danych tabelarycznych występują złożone zależności między wartościami atrybutów, często wykraczające poza proste korelacje. Zmiana wartości jednego atrybutu może wymagać proporcjonalnych modyfikacji w całej grupie powiązanych zmiennych. Na przykład, w danych kredytowych zmiana poziomu dochodu powinna być spójna nie tylko z poziomem wykształcenia, ale również z wiekiem, stażem pracy i sektorem zatrudnienia. Szczególnie istotne są korelacje wielowymiarowe, gdzie grupa zmiennych tworzy spójny profil, a naruszenie tej spójności może łatwo zdemaskować atak [15]. Rozważa się z tego powodu techniki *embedding-based perturbations*, które modyfikują dane na poziomie reprezentacji wektorowych, co pozwala zminimalizować nienaturalność zmian [4].

2.5.4 Minimalizacja liczby zmian wartości atrybutów

W odróżnieniu od danych obrazowych, gdzie zakłócenia mogą być rozproszone na wiele pikseli, w danych tabelarycznych ataki często koncentrują się na minimalizacji

liczby atrybutów, których wartości zostały zmienione. W niektórych przypadkach modyfikacja wartości pojedynczej, strategicznie wybranej zmiennej może być wystarczająca do skutecznego ataku. Funkcje odległości takie jak L_0 i odległość Hamminga są szczególnie przydatne do oceny takich zakłóceń [14].

2.5.5 Wrażliwość na systemy detekcji

W krytycznych zastosowaniach, takich jak finanse czy medycyna, modele przetwarzające dane tabelaryczne często są chronione przez zaawansowane systemy wykrywania anomalii. Dlatego ataki muszą być projektowane tak, aby unikać wzbudzania podejrzeń tych systemów. Skuteczne podejścia, jak PermuteAttack [32], wykorzystują algorytmy genetyczne do generowania zakłóceń trudnych do odróżnienia od naturalnych danych.

2.5.6 Specyfika modeli

Modele ML używane do przetwarzania danych tabelarycznych, takie jak XGBoost, CatBoost czy lasy losowe, różnią się od modeli neuronowych stosowanych w analizie obrazów. Szczególnym wyzwaniem są modele hybrydowe, łączące różne podejścia (np. drzewa decyzyjne z sieciami neuronowymi), które mogą wykazywać złożone wzorce podatności na ataki. Te modele często opierają się na nieciągłych granicach decyzyjnych [82].

2.5.7 Ataki na modele klasyfikujące dane tabelaryczne

Przy atakowaniu modeli klasyfikujących dane tabelaryczne rozróżnia się dwa kierunki - wykorzystywanie ataków heurestycznych, oraz dopasowanie do specyfiki danych tabelarycznych metod ogólnych.

Przykładem metody heurestycznej jest PermuteAttack [32], dedykowany danym tabelarycznym, w tym zawierającym zmienne kategoryczne. W trakcie implementacji ataku można zawrzeć znane ograniczenia dziedzinowe (np. wiek nieujemny, data ślubu późniejsza niż data urodzenia, itp.). W trakcie ataku dokonywana jest permutacja struktury danych - wartości cech lub zmiany ich kolejności - tak by zaburzyć ich interpretację przez model, zachowując semantykę i zgodność typów.

Kolejną kategorią ataków łatwo stosowalnych na danych tabelarycznych są ataki bazujące na próbkowaniu przestrzeni decyzyjnej. Nie wymagają one dostępu

do gradientów ani ciągłości przestrzeni decyzyjnej, są naturalnym wyborem przy atakowania modeli opartych o drzewa decyzyjne. Przykładem takiego ataku jest HopSkipJump [17], dla którego przy implementacji dla danych tabelarycznych konieczne jest tylko określenie dostępnych kategorii dla danych kategorycznych oraz ograniczenie zakresu zaburzeń do dopuszczalnych wartości. W celu zmniejszenia wykrywalności można też ograniczyć liczbę zmienianych jednocześnie cech.

Inną strategią jest użycie metody odtwarzającej lokalne gradienty funkcji straty. Najbardziej znaną z takich metod jest atak typu ZOO (Zeroth-Order Optimization) [18], który działa bezpośrednio na podstawie wartości funkcji celu. Mając przybliżone wartości gradientu, wybiera zmienne mające największy wpływ na predykcję modelu. Metoda wymaga dla każdego atakowanego modelu ograniczenia zmiany wartości cech do ich dopuszczalnych przedziałów (np. wartości numeryczne w przedziale lub tylko dostępne wartości kategoryczne).

Ataki gradientowe, takie jak FGSM (Fast Gradient Sign Method, zwana też metodą FGM) [27], PGD (Projected Gradient Descent) [52] czy BIM (Basic Iterative Method) [47], dostosowuje się do danych tabelarycznych poprzez modyfikację funkcji strat tak, aby uwzględniała specyfikę danych. W przypadku tabelarycznych cech kategorycznych, gradienty mogą być stosowane do określenia najbardziej podatnych cech, które następnie są zmieniane na inne wartości zgodne z możliwymi kategoriami. Metody gradientowe można też wykorzystać w przypadku pozyskania modelu zastępczego dla modelu atakowanego [46].

Ostatnią kategorią ataków na modele tabelaryczne są ataki generatywne, w których wykorzystuje się zdolność do generalizacji przejawianą przez duże modele językowe, w celu tworzenia przykładów adwersaryjnych zdolnych do zaburzenia działania modelu. Ten obszar badań jest jeszcze bardzo świeży, i na pewno przyniesie w najbliższej przyszłości interesujące rozwiązania [77].

3 Opis proponowanej metody wykrywania ataków

Podstawą zaproponowanej metody wykrywania ataków jest połączenie trzech komponentów analitycznych:

- Metody tworzenia modeli zastępczych zastępczych na podstawie obserwacji modelu monitorowanego.
- Analizy atrybutów diagnostycznych pozyskanych z modelu zastępczego.
- Stworzenia klasyfikatorów rozpoznających możliwość wystąpienia ataku dla konkretnej próbki danych, działających na podstawie atrybutów diagnostycznych.

Proponowana metoda opiera się na założeniu, że w trakcie przygotowania modelu zastępczego model obserwowany nie podlega atakom - uczymy się takich jego zachowań jakie występują w trakcie pierwotnej obserwacji. Do stworzenia modelu zastępczego wykorzystuję tutaj nowatorskie podejście oparte o metodykę BrightBox [41]. Bazując na przygotowanym modelu zastępczym, oraz obserwacji jego pracy (przetwarzania danych tych samych co model obserwowany) pozyskiwane są atrybuty diagnostyczne. Następnie, bazując na tych atrybutach, przygotowywane są klasyfikatory przetwarzające na bieżąco atrybuty diagnostyczne w celu wykrycia przypadku celowej zmiany wartości danych - ataku adwersaryjnego. Ataki odróżniane są nie na podstawie samych charakterystyk danych, ale na podstawie tego co dzieje się z tymi danymi w trakcie ich klasyfikacji - np. obserwowane są zmiany w charakterystyce otoczeń klasyfikowanych przykładów.

Metoda zakłada co najmniej możliwość wytrenowania modelu zastępczego dla nowego modelu objętego monitoringiem. Analiza atrybutów diagnostycznych oraz wytrenowanie klasyfikatorów można przeprowadzić dla każdego nowego modelu objętego

monitoringiem, przy czym wstępne wyniki wskazują iż nie jest to konieczne bez wyraźnej potrzeby - zmiany atrybutów diagnostycznych wykrywane przez klasyfikatory wskazują na ogólne cechy ataków.

3.1 Modele zastępcze

Metodyka tworzenia i wykorzystywania modeli zastępczych opiera się na zastosowaniu teorii zbiorów przybliżonych (ang. rough sets theory, RST) do aproksymacji działania złożonych, często nieprzejrzystych modeli uczenia maszynowego typu *czarna skrzynka*. Podejście to umożliwia szczegółową diagnozę działania modelu poprzez budowę interpretowalnych aproksymacji, zwanych modelami zastępczymi, które replikują predykcje oryginalnego modelu, zachowując jednocześnie możliwość interpretacji wyników. Tworzenie modeli zastępczych odbywa się w kilku etapach:

- dyskretyzacja atrybutów;
- konstrukcja przybliżonych reduktów;
- konstrukcja modelu zastępczego jako zespołu reduktów.

W niniejszej pracy zdecydowano się na zastosowanie metody budowy modelu zastępczego opartej na zespole reduktów aproksymacyjnych, ponieważ jest to podejście stabilne i odporne na drobne zmiany w danych wejściowych. Metody zespołowe, takie jak tu zastosowana, charakteryzują się wysoką stabilnością w sensie aproksymacji decyzji modelu bazowego, co jest kluczowe w kontekście detekcji ataków adwersaryjnych, gdzie nawet niewielkie zakłócenia w danych mogą znacząco wpływać na wyniki. Dodatkowo - istnieję dobre podstawy teoretyczne dla heurystyk i algorytmów probabilistycznych służących do wyprowadzania zespołów możliwie różnorodnych reduktów (tj. opartych na różnych atrybutach) na podstawie danych obserwacyjnych [40, 42, 53]. Wybór metody opartej o teorię zbiorów przybliżonych wynika przede wszystkim z jej właściwości merytorycznych - teoria RST została opracowana od samego początku jako narzędzie do pracy z wiedzą niepełną, w szczególności z pojęciami niewpewnymi i nie w pełni zdefiniowanymi. Charakteryzuje się tym, że analizowane są wyłącznie fakty ukryte w danych, nie są wymagane żadne dodatkowe informacje o danych, a po przeprowadzeniu analizy otrzymuje się rodzaj skondensowanej i interpretowalnej reprezentacji wiedzy. Istotny był również praktyczny aspekt wdrożeniowy - zgodność

i zdolności do integracji z innymi metodami i systemami analitycznymi rozwijanymi w QED Software. W szczególności dotyczy to z jej zgodności z systemami BrightBox [41][1] oraz InfoFrames[2].

3.1.1 Dyskretyzacja atrybutów

Przed przystąpieniem do budowy modelu zastępczego, atrybuty numeryczne w zbiorze danych są poddawane dyskretyzacji. Jest to o tyle istotne, że teoria zbiorów przybliżonych operuje zasadniczo na wartościach dyskretnych [62]. Realizuje się to metodą kwantylową, która dzieli zakres wartości numerycznych na przedziały zawierające mniej więcej taką samą liczbę obserwacji. Dla zbioru danych z atrybutami $A = \{a_1, ..., a_k\}$, gdzie każdy $a_i : U \to \mathbb{R}$ jest funkcją ciągłą określoną na uniwersum U, dyskretyzacja kwantylowa przekształca każdy a_i w funkcję dyskretną $a'_i : U \to \{v_1, ..., v_m\}$, gdzie m to liczba przedziałów, a v_j reprezentuje j-tą zdyskretyzowaną wartość. Dla każdego atrybutu a_i transformacja ta jest określona przez punkty podziału $\{p_{i,1}, ..., p_{i,m-1}\}$, takie że $|\{x \in U : p_{i,j-1} \leqslant a_i(x) < p_{i,j}\}| \approx |U|/m$ dla każdego j, gdzie $p_{i,0} = -\infty$ oraz $p_{i,m} = \infty$.

Metoda kwantylowa została wybrana specjalnie dla tego podejścia ze względu na jej efektywność obliczeniową oraz odporność. Choć inne metody dyskretyzacji mogą zapewnić bardziej precyzyjne przedziały dla poszczególnych atrybutów, zespołowy charakter podejścia opartego na reduktach (opisane w dalszej części pracy) skutecznie łagodzi potencjalne niedoskonałości prostszej dyskretyzacji. Ponieważ wiele reduktów wychwytuje różne perspektywy granic decyzyjnych, a ich predykcje są uśredniane, lokalne niedoskonałości w dyskretyzacji są zwykle niwelowane w skali całego zespołu. Efekt uśredniania, w połączeniu z liniową złożonością czasową metody oraz jej zdolnością do obsługi atrybutów o różnych skalach bez konieczności wstępnego przetwarzania, sprawia, że dyskretyzacja kwantylowa jest preferowaną metodą dyskretyzacji przy konstrukcji reduktów w celu wykorzystywania ich zespołów [20].

Wynikowa zdyskretyzowana przestrzeń umożliwia obliczenie relacji nierozróżnialności IND_B na U, gdzie $(x_i, x_j) \in \mathrm{IND}_B \iff \forall a \in B\, a'(x_i) = a'(x_j)$ dla dowolnego podzbioru $B \subseteq A$, co stanowi podstawę do dalszej analizy sąsiedztw oraz obliczania atrybutów diagnostycznych.

[1] https://qed.pl/nasze-rozwiazania/brightbox/
[2] https://qed.pl/nasze-rozwiazania/infoframes/

3.1.2 Konstrukcja przybliżonych reduktów

Podstawowym etapem konstrukcji modelu zastępczego jest stworzenie przybliżonych reduktów (ang. approximate reducts), czyli podzbiorów atrybutów, które zachowują prawie taką samą ilość informacji, jak pełny zbiór [73]. Użycie reduktów pozwala na upraszczenie modelu bez znacznej utraty dokładności. Konstrukcja reduktów przebiega zgodnie z algorytmem zaimplementowanym w pakiecie **RoughSets** [65]. Jest to algorytm przeszukiwania, w którym kolejne atrybuty dodawane są do reduktu na podstawie wybranej miary heurystycznej - zazwyczaj miary ważności (redukcji błędu klasyfikacji) oraz miary przyrostu informacji wnoszonej do atrybutu decyzyjnego, Wybór reduktów opiera się na z góry zdefiniowanym progu aproksymacji, oznaczanym jako ϵ, który określa poziom dokładności, jaki muszą osiągnąć redukty w stosunku do oryginalnego zbioru danych. Redukt uznaje się za przybliżony, jeśli zachowuje co najmniej $(1 - \epsilon)$ całkowitej informacji zakodowanej w pełnym zbiorze atrybutów. Redukty pozwalają na kompaktową reprezentację procesu decyzyjnego, odgrywając kluczową rolę w budowie interpretowalnych modeli zastępczych.

Redukty przybliżone stanowią podstawową koncepcję teorii zbiorów przybliżonych. Umożliwiają efektywne przybliżanie modeli przy jednoczesnym zachowaniu kluczowych właściwości decyzyjnych. Formalnie, dla danego uniwersum obiektów U, charakteryzowanego przez atrybuty A oraz miarę zależności funkcyjnej $\phi_d : 2^A \to \mathbb{R}$, opisującą zależności między podzbiorami atrybutów a decyzjami, (ϕ_d, ϵ)-przybliżony redukt $AR \subseteq A$ definiuje się jako nieredukowalny podzbiór atrybutów spełniający łącznie dwa warunki:

- $\phi_d(AR) \geqslant (1 - \epsilon)\phi_d(A)$, gdzie $\epsilon \in [0, 1)$ jest progiem aproksymacji;
- żaden właściwy podzbiór $AR' \subset AR$ nie spełnia pierwszego warunku.

Typowe wybory dla miary zależności ϕ_d obejmują miary takie jak wzajemna informacja (ang. mutual information), lub miara zanieczyszczenia Giniego (Gini impurity gain). Przybliżone redukty są konstruowane za pomocą heurystycznych algorytmów wyszukiwania, ponieważ znalezienie wszystkich możliwych reduktów jest problemem NP-trudnym. Obliczenia skupiają się na identyfikacji podzbiorów atrybutów, które przybliżają granicę decyzyjną, jednocześnie zachowując wykonalność obliczeniową.

Prawdopodobieństwa klas decyzyjnych oblicza się jako:

$$q_{ARi}(x) = \begin{cases} \frac{|[x]_{AR} \cap X_D \cap [\cdot]_{d_i}|}{|[x]_{AR} \cap X_D|} & \text{gdy } [x]_{AR} \cap X_D \neq \varnothing, \\ p_i & \text{w przeciwnym razie,} \end{cases}$$

gdzie $[\cdot]_{d_i}$ reprezentuje klasę równoważności decyzji d_i, a X_D to diagnozowany zbiór danych.

Parametr ϵ (epsilon) w (ϕ_d, ϵ)-przybliżonych reduktach reprezentuje próg jakości aproksymacji, który określa, jak blisko podzbiór atrybutów musi zachować zdolność dyskryminacyjną pełnego zbioru atrybutów. Formalnie, dla $\epsilon \in [0, 1)$, redukt AR musi spełniać warunek $\phi_d(AR) \geqslant (1 - \epsilon)\phi_d(A)$, gdzie ϕ_d mierzy zależność funkcyjną między atrybutami a decyzjami.

Praktyczne znaczenie ϵ można rozumieć jako tolerancję na utratę informacji: gdy $\epsilon = 0$, wymagamy dokładnych reduktów, które w pełni zachowują zdolność dyskryminacyjną oryginalnego zbioru atrybutów ($\phi_d(AR) = \phi_d(A)$), natomiast większe wartości ϵ pozwalają na większą utratę informacji w zamian za potencjalnie mniejsze podzbiory atrybutów. Na przykład, dla $\epsilon = 0.1$, redukt musi zachować co najmniej 90% wartości miary zależności pierwotnego zbioru. Jak wykazano w publikacjach [40] [42], wybór ϵ stanowi kluczowy kompromis: mniejsze wartości prowadzą do bardziej precyzyjnych, ale potencjalnie większych reduktów, podczas gdy większe wartości pozwalają na bardziej zwarte redukty kosztem zmniejszonej precyzji. Ten parametr, podobnie jak liczbą reduktów k, jest dostrajany za pomocą wyszukiwania (ang. grid search).

Warto podkreślić, że wartości decyzyjne, dla których konstruowane są przybliżone redukty odpowiadają predykcjom modelu diagnozowanego, tj. $M(X_D)$, a nie rzeczywistym wartościom docelowym (ang. ground truth).

3.1.3 Model zastępczy jako zespół reduktów

Zamiast polegać na pojedynczym redukcie, model zastępczy używa zespołu reduktów. Takie podejście zwiększa zdolność modelu zastępczego do dokładnej aproksymacji działania modelu *czarnej skrzynki*, poprzez łączenie decyzji podejmowanych na podstawie wielu reduktów. Każdy redukt wnosi swoją część do ostatecznej predykcji, a wynik końcowy uzyskuje się poprzez uśrednienie wyników wszystkich reduktów.

Zbiór reduktów stanowi podstawę struktury modelu zastępczego, łącząc wiele przybliżonych reduktów w celu osiągnięcia pewnej i dokładnej aproksymacji modelu

obserwowanego. Zbiór

$$R = \{AR_1, \ldots, AR_k\},$$

zazwyczaj składający się z $k \approx 2500$ reduktów, działa jako estymator bootstrapowy (ang. bagging estimator), gdzie każdy redukt AR_i zapewnia inną perspektywę na granice decyzyjne.

Zespół reduktów działa jako model zastępczy dzięki mechanizmowi głosowania ważonego prawdopodobieństwem. Dla danego obiektu $x \in U$ oraz zespołu reduktów $R = \{AR_1, ..., AR_k\}$, proces klasyfikacji przebiega w następujących krokach:

- Obliczanie prawdopodobieństw klas decyzyjnych:

  Dla każdego reduktu $AR \in R$, prawdopodobieństwa klas decyzyjnych $q_{AR}(x) = (q_{AR1}(x), ..., q_{ARl}(x))$ są obliczane jako:

  $$q_{AR}(x) = \frac{1}{|[x]_{AR}|} \sum_{(x_i, d_i) \in [x]_{AR}} \text{onehot}_l(d_i),$$

  gdzie $[x]_{AR}$ oznacza klasę nierozróżnialności x, zdefiniowaną przez atrybuty zawarte w AR, a $\text{onehot}_l(d_i)$ reprezentuje jednowymiarowy wektor zakodowany dla decyzji d_i w l klasach. Jeśli $[x]_{AR} = \varnothing$, wówczas stosowany jest wcześniej używany rozkład prawdopodobieństw $p = (p_1, ..., p_l)$.

- Agregacja zespołu reduktów:

  Zespół agreguje przewidywania indywidualnych reduktów za pomocą uśredniania:

  $$q_R(x) = \frac{1}{|\{AR \in R : [x]_{AR} \neq \varnothing\}|} \sum_{AR \in R, [x]_{AR} \neq \varnothing} q_{AR}(x),$$

  Uśrednienie zwiększa odporność na błędy poszczególnych reduktów i wykorzystuje jedynie te, które generują istotne przewidywania.

- Końcowa decyzja klasyfikacyjna:

  Klasa z maksymalnym zagregowanym prawdopodobieństwem zostaje wybrana jako wynik klasyfikacji:

  $$\bar{M}(x) = \arg\max_i q_{R_i}(x).$$

  Jeśli żaden z reduktów nie zapewnia ważnych przewidywań ($[x]_{AR} = \varnothing$ dla wszystkich $AR \in R$), model stosuje rozkład domyślny p.

- Ocena jakości i optymalizacja modelu zastępczego:

  Jakość przybliżenia modelu diagnozowanego modelem zastępczym weryfikowana jest za pomocą współczynnika kappa Cohena, gdzie zakłada się że ($\kappa > 0.9$ wskazuje na udane przybliżenie). Jeśli jakość aproksymacji nie osiąga zadanego progu, proces konstrukcji modelu zastępczego jest powtarzany przy zmienionych wartościach parametrów wpływających na jakość modelu zastępczego - zmieniana jest m.in. liczba przedziałów dyskretyzacji, liczba reduktów aproksymacyjnych oraz ich dokładność.

Podejście zespołowe wykazuje rosnącą stabilność wraz ze zwiększającą się liczbą reduktów, co mierzono zarówno poprzez redukcję liczby pustych sąsiedztw, jak i zbieżność podobieństwa Jaccarda sąsiedztw między kolejnymi partiami reduktów. Stabilność ta jest szczególnie istotna w przypadku dużych zbiorów danych. Przykładowo w [41] pokazano, jak dla zbiorów zawierających około 59 427 instancji i 455 atrybutów, stabilne sąsiedztwa i spójne aproksymacje osiągnięto przy standardowym rozmiarze zespołu wynoszącym 2500 reduktów.

3.2 Atrybuty diagnostyczne

Jedną z kluczowych cech modelu zastępczego jest zdolność do obliczania i analizowania sąsiedztw instancji danych. Dla obiektu $x \in U$ sąsiedztwo $N(x)$ definiuje się jako zbiór podobnych obiektów w zbiorze treningowym X_R, określony przez klasę nierozróżnialności $[x]_{AR}$ dla każdego przybliżonego reduktu AR. Sąsiedztwo składa się z obserwacji z referencyjnego zbioru danych, które są podobne do diagnozowanej instancji, zgodnie z zespołem reduktów. Podobieństwo definiuje się na podstawie liczby reduktów, dla których dwie instancje mają te same kombinacje wartości atrybutów. Sąsiedztwo odgrywa kluczową rolę w diagnozowaniu działania modelu *czarnej skrzynki*, ponieważ dostarcza lokalnego kontekstu do oceny decyzji modelu. Analizując spójność predykcji w sąsiedztwie – na przykład czy podobne instancje zostały sklasyfikowane poprawnie lub błędnie – model zastępczy może dostarczyć wgląd w przyczyny błędów modelu.

Po obliczeniu sąsiedztw, model zastępczy uzupełnia diagnozę poprzez wyliczenie zestawu atrybutów diagnostycznych. Atrybuty diagnostyczne dostarczają ustandaryzowanego zestawu miar umożliwiających analizę zachowania modelu poprzez charakterystyki sąsiedztw. Dla obiektu $x \in X_D$, którego sąsiedztwo $N(x) \subseteq X_R$

określono na podstawie relacji nierozróżnialności wyznaczonych przez przybliżone redukty, obliczane są następujące atrybuty diagnostyczne:

- **Rozmiar sąsiedztwa** (ang. neighborhood size) – liczba obserwacji znajdujących się w sąsiedztwie diagnozowanej obserwacji, znormalizowana jako $|N(x)|/|X_R|$. Mierzy, jak duże jest sąsiedztwo dla każdej instancji, odzwierciedlając liczbę podobnych instancji znalezionych w zbiorze referencyjnym.

- **Niepewność** (ang. uncertainty) – miara niepewności predykcji, obliczana na podstawie znormalizowanej entropii:

$$H_{\text{norm}}(q(x)) = -\sum_{i=1}^{l} q_i(x) \log(q_i(x)) / \log(l),$$

  gdzie l to liczba klas decyzyjnych, a $q_i(x)$ to prawdopodobieństwo klasy i dla obserwacji x. Jest to krytyczny atrybut diagnostyczny, mierzący pewność modelu zastępczego co do swoich predykcji. Niska niepewność może wskazywać na nadmierną pewność co do błędnych predykcji, podczas gdy wysoka niepewność sugeruje trudności modelu w prawidłowej predykcji.

- Metryki spójności: te atrybuty mierzą, jak często predykcje dla instancji w sąsiedztwie zgadzają się z wartościami docelowymi lub oryginalnymi predykcjami modelu. Pomagają one określić, czy błędy modelu *czarnej skrzynki* są systematyczne, czy losowe. Podstawowe metryki spójności to:

  - **Spójność celu z przybliżeniami w sąsiedztwie** (ang. target consistency with approximations in neighborhood) – miara określająca zgodność wartości celu diagnozowanej obserwacji d_x z przybliżonymi wartościami predykcji $\bar{M}(N(x))$ uzyskanymi dla obserwacji znajdujących się w jej sąsiedztwie $N(x)$.

    Obliczana jako

$$\text{TC}(x) \;=\; \frac{1}{|N(x)|} \sum_{z \in N(x)} \mathbf{1}\Big(\arg\max_{c} \; P_{\overline{M}}\big(c \mid z\big) \;=\; d(x)\Big),$$

    gdzie: $P_{\overline{M}}\big(c \mid z\big)$ to prawdopodobieństwo (wg modelu $\overline{M}$) przypisania klasy c dla punktu z, a $\mathbf{1}(\cdot)$ to funkcja wskaźnikowa, przyjmująca wartość 1, gdy spełniony jest warunek w nawiasie, w przeciwnym razie 0.

– **Spójność predykcji z celami w sąsiedztwie** (ang. prediction consistency with targets in neighborhood) – miara określająca zgodność predykcji $M(x)$ dla diagnozowanej obserwacji x z wartościami celu $d_{N(x)}$ obserwacji należących do jej sąsiedztwa $N(x)$. Obliczana jako:

$$\mathrm{PC}(x) \;=\; \frac{1}{|N(x)|} \sum_{z \in N(x)} \mathbf{1}\Big(M(x) = d(z)\Big),$$

gdzie $d(z)$ to rzeczywista wartość celu (etykieta) obserwacji z, a $\mathbf{1}(\cdot)$ to funkcja wskaźnikowa, przyjmująca wartość 1, gdy warunek jest spełniony, w przeciwnym razie 0.

– **Spójność celu z celami w sąsiedztwie** (ang. target consistency with targets in neighborhood) – miara opisująca zgodność wartości celu diagnozowanej obserwacji d_x z wartościami celu $d_{N(x)}$ obserwacji znajdujących się w jej sąsiedztwie $N(x)$.

Obliczana jako:

$$\mathrm{TC}_{\mathrm{targets}}(x) \;=\; \frac{1}{|N(x)|} \sum_{z \in N(x)} \mathbf{1}\Big(d(x) \;=\; d(z)\Big),$$

– **Niespójność celów i przybliżeń w sąsiedztwie** (ang. targets and approximations inconsistency in neighborhood) – miara określająca stopień niespójności pomiędzy wartościami celu $d_{N(x)}$ a przybliżeniami $q_{d_{x'}}(x')$ uzyskanymi dla obserwacji $x' \in N(x)$. Obliczana jest jako:

$$\frac{1}{|N(x)|} \sum_{x' \in N(x)} \left[1 - q_{d_{x'}}(x')\right],$$

gdzie $q_{d_{x'}}(x')$ to przybliżone prawdopodobieństwo klasy decyzyjnej $d_{x'}$ dla obserwacji x'.

– **Różnorodność celów w sąsiedztwie** (ang. targets diversity in neighborhood) – miara opisująca zróżnicowanie wartości celu w sąsiedztwie diagnozowanej obserwacji w porównaniu z różnorodnością wartości celu obliczoną dla całego zbioru danych diagnozowanych. Obliczana jako:

$$h(p, p(N(x))) = \frac{H(p)}{H(p, p(N(x)))},$$

gdzie p to rozkład aprioryczny, a $p(N(x))$ to rozkład klas decyzyjnych w sąsiedztwie $N(x)$.

– **Różnorodność przybliżeń w sąsiedztwie** (ang. approximations diversity in neighborhood) – miara opisująca zróżnicowanie wartości przybliżonych predykcji w sąsiedztwie diagnozowanej obserwacji w odniesieniu do zróżnicowania przybliżeń obliczonych dla całego zbioru diagnozowanych danych. Definiowana jako:

$$h(p, q(N(x))),$$

gdzie $q(N(x))$ to rozkład przybliżonych predykcji w sąsiedztwie $N(x)$.

Dla obiektów z pustymi sąsiedztwami wartości atrybutów są domyślnie określane na podstawie metryk opartych na rozkładach apriorycznych p. Znormalizowana entropia używana w obliczeniach niepewności jest zdefiniowana jako:

$$H_{\text{norm}}(q(x)) = -\sum_{i=1}^{l} q_i(x) \log(q_i(x)) / \log(l),$$

gdzie l to liczba klas decyzyjnych.

3.2.1 Interpretacja atrybutów diagnostycznych

Atrybuty diagnostyczne umożliwiają kompleksowy wgląd w działanie modelu, pomagając zidentyfikować potencjalne problemy, takie jak nadmierne dopasowanie, niedopasowanie czy też błędy wynikające z niejednorodności danych treningowych. Atrybuty te służą następnie do oceny spójności predykcji modelu. Na przykład, jeśli instancja otoczona jest sąsiadami z tymi samymi prawdziwymi etykietami, a model błędnie przewiduje jej etykietę z wysoką pewnością, może to wskazywać na problem nadmiernego dopasowania (overfitting). Z kolei, gdy model wykazuje dużą niepewność przy przewidywaniu etykiety dla instancji z dużym, spójnym sąsiedztwem, może to sugerować niedopasowanie modelu (underfitting). Analiza działania modelu zastępczego może także pomóc w globalnej ocenie działania modelu badanego, klasyfikując go jako nadmiernie dopasowany, niedopasowany lub właściwie dopasowany, na podstawie zagregowanych wartości atrybutów diagnostycznych dla całego zbioru danych. Traktowane łącznie, atrybuty diagnostyczne dostarczają wglądu w:

- wiarygodność modelu w różnych obszarach przestrzeni cech;
- obszary, w których model może popełniać systematyczne błędy;
- potencjalne wartości odstające lub nietypowe przypadki;

- regiony przestrzeni cech, w których zachowanie modelu może być niestabilne;
- ogólną odporność przewidywań modelu.

Lokalnie atrybuty diagnostyczne pomagają w zrozumieniu indywidualnych błędnych klasyfikacji, wskazując, czy wynikają one z nietypowych instancji (outliers) czy z systematycznych błędów w procesie uczenia modelu [41].

W zakresie niniejszej pracy leżała weryfikacja hipotezy mówiącej że analiza atrybutów diagnostycznych może wskazać istotne prawdopodobieństwo wystąpienia modyfikacji danych wejściowych noszącej znamiona modyfikacji celowej (ataku adwersaryjnego). Hipoteza ta została potwierdzona w sekcji 4.2.3.

3.3 Konstrukcja klasyfikatorów

Ostatnim elementem potrzebnym do stworzenia pełnego przebiegu diagnostycznego są właściwe klasyfikatory rozpoznające ataki.

Uwaga - w procedurze występują dwa rodzaje modeli klasyfikacyjnych. Pierwszy rodzaj klasyfikatorów to różne modele uczenia maszynowego działające na danych benchmarkowych. Celem ich stworzenia jest zasymulowanie normalnych działających modeli uczenia maszynowego, takich jakie mogą stać się celem ataków. W niniejszej pracy korzystamy ze sprawdzonych metod tworzenia tych modeli. W dalszych krokach procedury, bazując na obserwacji ich zachowania - zarówno w trakcie pracy na danych poprawnych jak i zaatakowanych - tworzone są modele klasyfikujące, mające za zadanie wykrywanie ataków. Stanowią one główny praktyczny wynik procedury. Wejściem do tych klasyfikatorów są wartościach atrybutów diagnostycznych oraz etykietach decyzyjnych wskazujących na obecność ataku lub jego brak.

Szczegółowa procedura ich przygotowania, bazująca na wcześniej opisanych krokach, przebiega następująco:

- Przygotowanie danych:
  - Przygotowanie benchmarkowych zbiorów danych klasyfikacyjnych - najlepiej różnorodnych i zawierających reprezentatywną próbkę różnych typów danych przetwarzanych.
  - Podział każdego zbioru na część treningową, służącą do uczenia klasyfikatorów oraz modeli zastępczych, i diagnostyczną, służącą do symulacji ataków.

  - W przypadku trenowania klasyfikatora dedykowanego pod konkretne zadanie (model/typ danych), istotne jest zapewnienie by dane w zbiorze treningowym były pozbawione przypadków ataków.

- Wytrenowanie różnych typów modeli na każdym zbiorze danych.

  W przypadku, gdy naszym celem jest stworzenie klasyfikatora wykrywającego ataki dedykowanego do diagnostyki i monitoringu konkretnego modelu ML, różnorodność modeli może zostać ograniczona do modeli z rodziny modelu docelowego.

- Konstrukcja modelu zastępczego:

  Dla każdej pary model-zbiór danych:

  - Dyskretyzacja atrybutów numerycznych metodą kwantylową [20], tak by każdy z przedziałów zawierał zbliżoną liczbę instancji.
  - Konstrukcja zespołu przybliżonych reduktów. W ramach procedury - przeszukiwanie przestrzeni parametrów: ϵ odpowiedzialnego za jakość przybliżenia reduktów oraz parametru określającego liczbę reduktów w zespole.
  - Walidacja jakości modelu zastępczego przy użyciu współczynnika kappa Cohena (cel > 0.9). Jeśli pożądana jakość nie może zostać osiągnięta, wynikiem jest zespół reduktów z ustawieniami hiperparametrów zapewniającymi najwyższą możliwą jakość przybliżenia - do dalszej decyzji eksperckiej czy zostanie on uznany za wystarczająco dobry.

- Generowanie ataków adwersaryjnych: Zastosowanie różnych znanych metod ataku na część diagnostyczną każdego zbioru. W niniejszej pracy metodę przetestowano przy użyciu trzech oraz sześciu różnych ataków.

- Obliczanie atrybutów diagnostycznych:

  Dla każdej instancji danych, obliczenie:

  - Metryk opartych na sąsiedztwie:
    * miar spójności celów;
    * miar spójności predykcji;
    * miar różnorodności;

        * rozmiaru sąsiedztwa.
    - Miar niepewności.

    W tym kroku można wykonać dodatkową walidację metody poprzez porównanie i analizę statystyczną rozkładów atrybutów diagnostycznych między instancjami oryginalnymi a zaatakowanymi.

- Trenowanie klasyfikatorów wykrywających ataki : Bazując na zagregowanych atrybutach diagnostycznych, trenowany jest klasyfikator wykrywający obecność/brak ataku dla danego zestawu danych (klasyfikacja binarna). W dotychczasowych eksperymentach wystarczającą jakość klasyfikacji osiągnięto z użyciem klasyfikatorów opartych o Lasy Losowe (ang. random forest) oraz XGBoost.

  Dodatkową zmienną, którą można wziąć pod uwagę jako wejście do klasyfikatora, może być miara jakości działania modelu diagnozowanego - np. miara balanced accuracy. Jej wykorzystanie zależne jest od konkretnego przypadku.

  Opcjonalnie - można pokusić się o wytrenowanie klasyfikatora izolującego ataki - rozróżniającego typy ataków - bazującego na tych samych cechach co klasyfikator wykrywający ataki. Tego typu klasyfikator jest mniej użyteczny niż klasyfikator wykrywający ataki, oraz posiada podstawową słabość polegająca na niezdolności do izolacji ataków nieznanych, niemniej możliwe do wyobrażenia są zastosowania w których próba jego wytrenowania będzie uzasadniona.

- Walidacja: Walidacja procesu trenowania poprzez skorzystanie z różnych scenariuszy ewaluacji:
    - walidacja krzyżowa;
    - wyłączenie jednego zbioru danych;
    - wyłączenie jednego modelu;
    - wyłączenie jednego ataku (tylko dla klasyfikatorów wykrywających ataki).

Takie podejście umożliwia stworzenie klasyfikatorów wykazujących silne zdolności detekcyjne oraz zdolność do generalizacji, co zostało wykazane w sekcji 4.

Dzięki oparciu klasyfikatorów o cechy uniwersalne atrybutów diagnostycznych, możliwe jest przenoszenie klasyfikatorów pomiędzy środowiskami diagnostycznymi. Możliwe jest również douczanie i przeuczanie ich przy każdej implementacji - tak by były dostrojone do specyficznych wymogów monitorowanych modeli uczenia maszynowego i przetwarzanych przez nie danych.

W niniejszej pracy potwierdzono skuteczność klasyfikatorów wykrywających ataki na poziomie balanced accuracy ~90-95%.

Dla klasyfikatorów rozróżniających ataki, najwyższa osiągnięta dotychczas jakość rozróżniania to ~75-80% balanced accuracy.

3.4 Przebieg procesu diagnostycznego

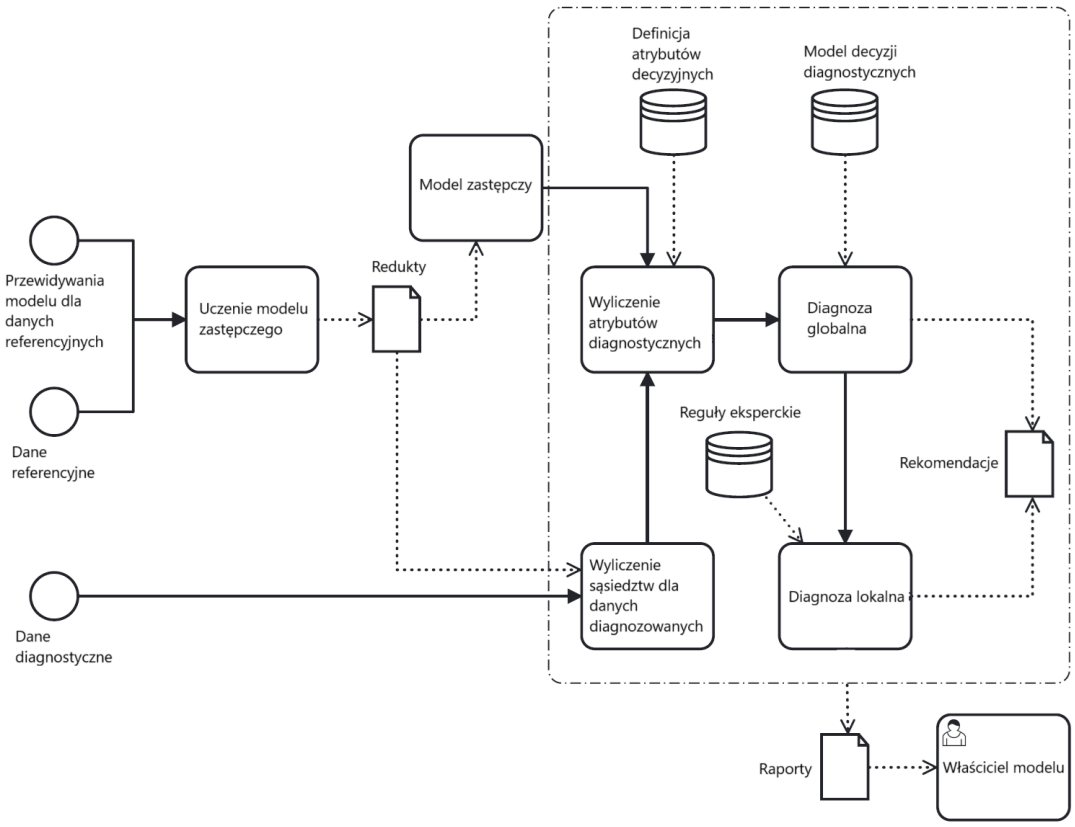
Bazując na opisanych powyżej komponentach, w niniejszej pracy badam metodę wykrywania ataków bazującą na następującym schemacie:

- Faza wstępna - badanie normalnego trybu działania modelu uczenia maszynowego - obserwacja wejść i wyjść z modelu, stworzenie na tej podstawie modelu zastępczego.
- Faza monitoringu - równoległe procesowanie danych przetwarzanych przez model wejściowy przez model zastępczy. Na tej podstawie - obserwacja stanu atrybutów diagnostycznych generowanych przez model zastępczy. Bazując na wartościach atrybutów zagregowanych dla danej próbki danych - wykrywanie, z użyciem stworzonego klasyfikatora - przypadków podejrzanych o celową modyfikację danych.
- Faza aktualizacji klasyfikatora - aktualizacja klasyfikatora bazująca na wiedzy o dostępnych nowych atakach, oraz o zidentyfikowanych przypadkach niepoprawnej klasyfikacji. W ramach analizy prowadzonej w trakcie aktualizacji modelu możliwe jest dokonanie zmiany zestawu cech wejściowych (feature engineering) wynikająca z reakcji na wystąpienie nowych typów ataku lub nowych cech modeli diagnozowanych.

3.4.1 Faza monitoringu

W trakcie fazy monitoringu stworzony zespół reduktów otrzymuje te same dane wejściowe co model obserwowany. Poniżej schemat procesu podejmowania decyzji przez zespół reduktów stanowiący model zastępczy.

- Dla instancji $x \in U$, każdy redukt AR w zespole $R = \{AR_1, \ldots, AR_k\}$:
  - Wyznacza klasę nierozróżnialności $[x]_{AR}$ - instancje, które mają te same wartości atrybutów w AR.



Rys. 3.1: Przebieg monitoringu modeli uczenia maszynowego z wykorzystaniem metody BrightBox [41].

– Oblicza prawdopodobieństwa klas dla tego reduktu jako:
$q_{AR}(x) = (q_{AR1}(x), \ldots, q_{ARl}(x))$, gdzie
$$q_{ARi}(x) = \begin{cases} \frac{|[x]_{AR} \cap X_D \cap [\cdot]_{d_i}|}{|[x]_{AR} \cap X_D|} & \text{jeśli } [x]_{AR} \cap X_D \neq \varnothing, \\ p_i & \text{w przeciwnym razie.} \end{cases}$$

– Końcowy rozkład prawdopodobieństwa jest obliczany poprzez uśrednienie po wszystkich odpowiednich reduktach:
$$q_R(x) = \frac{1}{|\{AR \in R : [x]_{AR} \cap X_D \neq \varnothing\}|} \sum_{AR \in R : [x]_{AR} \cap X_D \neq \varnothing} q_{AR}(x).$$

– Ostateczna decyzja klasyfikacyjna jest podejmowana przez wybór klasy o najwyższym prawdopodobieństwie:
$$\bar{M}(x) = \arg\max_i q_{R_i}(x).$$

– Jeśli $[x]_{AR} \cap X_D = \varnothing$ dla wszystkich $AR \in R$, klasyfikator domyślnie używa rozkładu a priori $p = (p_1, \ldots, p_l)$.

To podejście zespołowe działa jako probabilistyczny system głosowania, w którym:

- Każdy redukt reprezentuje inną perspektywę na granice decyzyjne.
- Głos każdego reduktu jest ważony przez prawdopodobieństwa, które przypisuje klasom.
- Prawdopodobieństwa ze wszystkich reduktów są uśredniane, łagodząc wpływ lokalnych niespójności.
- Klasa o najwyższym średnim prawdopodobieństwie jest wybierana jako końcowa predykcja.

Następnie, dla każdej instancji $x \in X_D$, obliczane jest jej sąsiedztwo $N(x) \subseteq X_R$. Na podstawie analizy sąsiedztwa, wyliczane są atrybuty diagnostyczne charakteryzujące każdą instancję.

Następnie dokonywana jest agregacja atrybutów dla ustalonej wielkości próbki danych. Wyliczane są statystyki podsumowujące cechy atrybutów diagnostycznych:

- średnia, minimum, maksimum;
- dolny kwartyl, mediana, górny kwartyl;
- rozstęp (ang. range).

Dodawane są dodatkowe charakterystyki:

- liczba obserwacji;
- liczba klas;
- zrównoważona dokładność (ang. balanced accuracy) wyników klasyfikacji - jeśli jest dostępna.

Powyższe zagregowane atrybuty traktowane są jako zbiór cech służących klasyfikacji - wykrywaniu ataków. Ich podzbiór, zdefiniowany na etapie wstępnego trenowania klasyfikatora, wprowadzany jest jako wektor danych wejściowych do klasyfikatora binarnego, udzielającego odpowiedzi na pytanie ”czy w danej próbce danych wystąpiły cechy wskazujące na wystąpienie ataku”. W przypadku gdy jest

to możliwe - zagregowane atrybuty diagnostyczne są wykorzystywane również jako wejście do klasyfikatora izolującego ataki - (klasyfikacja wieloklasowa: rozróżnianie typów ataków).

W praktycznym zastosowaniu kluczowe jest ustalenie okna dla próbki danych dla których dokonywana jest agregacja i następująca po niej detekcja. W niniejszej pracy testowane było okno o rozmiarze nie mniejszym niż 100 punktów danych. W zastosowaniu rzeczywistym wielkość ta stanowi parametr powiązany z wydajnością systemu oraz maksymalnym akceptowalnym opóźnieniem detekcji.

4 Przebieg eksperymentów

Celem prowadzonych prac było stworzenie metody, której konkretyzacją jest klasyfikator umożliwiający wykrywanie ataków na modele ML dedykowane dla rozwiązywania problemów klasyfikacji i trenowanych na danych tabelarycznych. Zgodnie z przyjętymi założeniami metoda ma jedynie dostęp do bieżących i historycznych decyzji podejmowanych przez diagnozowany model ML oraz do niezaburzonych danych treningowych, na których trenowana była pierwotna - a więc z założenia nie poddana atakowi - wersja metody.

Prace badawcze podzielone zostały na następujące etapy:

- Przygotowanie modeli uczenia maszynowego działających na publicznie dostępnych zbiorach danych. W tym kroku celem był wybór różnorodnych zbiorów danych oraz co najmniej trzech różnych typów modeli uczenia maszynowego innych niż sieci neuronowe.

- Implementacja wybranych metod ataków. Do badan wybrano ataki charakteryzujące się niskim profilem działania (stosowaniem najmniejszych perturbacji służących osiągnięciu celu ataku), dostępnością opisu literaturowego oraz implementacji, a także względną nowością.

- Analiza, z wykorzystaniem modeli zastępczych, sposobu działania modeli uczenia maszynowego w trybie bez ataku oraz z atakiem. Wynikiem tego etapu jest zestaw atrybutów diagnostycznych generowanych przez modele zastępcze w stanie bez ataku oraz przy zadanym ataku.

- Analiza statystyczna wartości atrybutów diagnostycznych - stwierdzenie czy występują istotne różnice pomiędzy stanami z atakiem oraz bez ataku, jak też określenie wagi poszczególnych atrybutów diagnostycznych i ich istotności ze względu na wykrywanie ataku.

- Stworzenie klasyfikatora umożliwiającego wykrywanie stanu podejrzenia o atak. Klasyfikator operuje m.in. na wartościach atrybutów diagnostycznych.

- Rozszerzenie badań na ataki celujące w modele oparte o sztuczne sieci neuronowe.
- Weryfikacja działania stworzonego klasyfikatora w funkcji parametru określającego intensywności ataku. Stworzony wcześniej klasyfikator, bazujący na najczęstszych parametrach ataków, zostanie sprawdzony w wybranych scenariuszach w których intensywność ataku jest przeskalowana. Odpowiada to sytuacji w której atakujący przeprowadza ataki o różnych celach - odpowiada to zróżnicowaniu celu skali ataku pomiędzy niewielkim a pełnym zaburzeniem działania modelu uczenia maszynowego.

4.1 Opis danych i środowiska eksperymentalnego

4.1.1 Dane

Eksperymenty prowadzone były na zestawie 22 zbiorów danych tabelarycznych, udostępnionych w serwisie OpenML[1], przygotowanych pod zadania klasyfikacji. Wybrane zostały zbiory danych dla których wiadome było, że możliwe jest zbudowanie poprawnie działających klasyfikatorów - np. w związku z wykorzystywaniem ich w konkursach data science na platformie Kaggle[2] i podobnych.

Uzasadnienie wyboru zbiorów danych

Przy wyborze konkretnych zbiorów danych kierowano się celem, którym było sprawdzenie skuteczności i uniwersalności metody wykrywania ataków w różnych scenariuszach oraz przy zróżnicowanych cechach danych. Dane powinny zapewniać różnorodność i reprezentatywność typowych problemów opisywanych przez dane tabelaryczne, i pozwalać na kompleksową ocenę metody w różnych kontekstach i na różnorodnych oraz przydatnych danych. Poniżej opisujemy powody, dla których wybrano te konkretne zestawy danych.

- Różne dziedziny i zastosowania

  Zbiory danych pochodzą z różnych dziedzin, takich jak bioinformatyka (Bioresponse), finanse (Churn), medycyna (WDBC), środowisko (Wilt), nauki przyrodnicze (QSAR-Biodeg) i rozpoznawanie obrazów (Mfeat-Fourier). Dzięki

[1] https://www.openml.org/
[2] https://www.kaggle.com/

Tabela 4.1: Zbiory danych użyte w eksperymentach. Kolumny N, $|A|$ i $|L|$ zawierają informacje o: liczbie przykładów, liczbie atrybutów oraz liczbie klas.

Nazwa	N	$\|A\|$	$\|L\|$
Bioresponse	3751	1776	2
churn	5000	20	2
cmc	2000	47	10
cnae-9	1080	856	9
dna	3186	180	3
har	10299	561	6
madelon	2600	500	2
mfeat-factors	2000	47	10
mfeat-fourier	2000	76	10
mfeat-karhunen	2000	47	10
mfeat-zernike	2000	47	10
nomao	34465	118	2
optdigits	2000	47	10
pendigits	10992	16	10
phoneme	5404	5	2
qsar-biodeg	1055	41	2
satimage	6430	36	6
semeion	1593	256	10
spambase	2000	47	10
wall-robot-navigation	5456	24	4
wdbc	569	30	2
wilt	4839	5	2

temu, że eksperymenty prowadzone były na szerokim zakresie problemów, możliwa jest ocena testowanej metody wykrywania ataków pod kątem jej elastyczność i przydatność w różnych zastosowaniach.

- Zróżnicowane cechy danych
  - Wymiarowość: Rozpiętość liczby cech obejmuje od 5 (Phoneme, Wilt) do 1776 cech (Bioresponse). Umożliwia to ocenę metody zarówno na zbiorach o niskiej, jak i wysokiej wymiarowości, dostarczając wniosków na temat jej skalowalności i efektywności.
  - Rozmiar: Liczba próbek (przykładów) waha się od mniejszych zbiorów, liczących mniej niż 1000 przykładów (WDBC, Semeion), do dużych zbiorów z dziesiątkami tysięcy rekordów (Nomao, Wall-Robot-Navigation). Dzięki temu możliwa jest ocena efektywności obliczeniowej metody na różnych skalach danych.
  - Trudność zadania: Zbiory obejmują różne problemy, od prostych zadań binarnych (np. Churn) po wieloklasowe klasyfikacje (np. Pendigits i Satimage). Sprawdzamy, jak metoda radzi sobie w trudniejszych przypadkach.
- Dane tabelaryczne

  Wybrane zbiory reprezentują typowe cechy danych tabelarycznych, takie jak współwystępowanie cech numerycznych, kategorycznych czy mieszanych. Przykładowo, HAR i Satimage zawierają ciągłe dane sensoryczne, a Churn i CNAE-9 cechy kategoryczne i przetworzone dane tekstowe. Nomao i Churn umożliwiły testy na większych, realistycznych zbiorach danych o mieszanych typach cech. Taki dobór zbiorów pozwalał na ocenę czy metoda dobrze działa w kontekście struktury danych tabelarycznych.
- Dostępność danych benchmarkowych

  Wszystkie wybrane zbiory danych są dobrze ugruntowanymi benchmarkami, co zapewnia porównywalność z wcześniejszymi badaniami. Są one szeroko cytowane w literaturze i publicznie dostępne na platformach takich jak OpenML czy Kaggle. Dzięki temu są wiarygodnymi punktami odniesienia i oceny porównawczej.
- Przydatność w testach adwersaryjnych

Badana metoda została przygotowana z myślą o detekcji ataków adwersaryjnych. Mając to na uwadze, zbiory zostały dobrane tak, aby uwzględniać zarówno dane syntetyczne jak i rzeczywiste. Dzięki temu możliwe jest realistyczne modelowanie różnych scenariuszy. Przykładowo:

  - Madelon to zbiór syntetyczny zaprojektowany specjalnie do testowania metod selekcji cech i stanowi bazę do badania zakłóceń w przestrzeniach wysokowymiarowych.
  - Rzeczywiste zbiory, takie jak Spambase i Phoneme, reprezentują dziedziny podatne na ataki adwersaryjne, takie jak wykrywanie spamu i przetwarzanie mowy, gdzie drobne zakłócenia mogą znacząco wpłynąć na wyniki modeli.

- Zróżnicowane zadania

  Zbiory reprezentują różne typy problemów uczenia maszynowego, takie jak klasyfikacja binarna (Bioresponse, WDBC), klasyfikacja wieloklasowa (Pendigits, Optdigits) i zadania predykcyjne z cechami ciągłymi (Wall-Robot-Navigation).

- Wartość naukowa

  Niektóre zbiory danych oferują dodatkowe korzyści naukowe, np.:

  - HAR i Satimage pozwalają ocenić, jak metoda radzi sobie z czasowymi i przestrzennymi zależnościami w danych.
  - CNAE-9 i Mfeat umożliwiają testy na danych o silnie skorelowanych cechach.

4.1.2 Modele atakowane

Dobór metod uczenia maszynowego mających być celami ataków został przeprowadzony pod kątem zróżnicowanie typów modeli, tak by umożliwić ocenę skuteczności proponowanych metod wykrywania i izolacji ataków w różnych warunkach. Wybrano szereg klasyfikatorów reprezentujących różne paradygmaty uczenia maszynowego, w tym metody liniowe oraz metody zespołowe. Każdy z tych modeli wprowadza różne granice decyzyjne, strategie uczenia i podatności na ataki. Stanowią też przykłady modeli często występujących w praktycznych implementacjach systemów automatyzujących decyzje opartych o uczenie maszynowe.

Do celów ataków w podstawowym eksperymencie wybrano następujące modele uczenia maszynowego:

- regresja logistyczna;
- maszyna wektorów nośnych (SVM);
- algorytm XGBoost

Regresja logistyczna pełni rolę prostego modelu bazowego, szczególnie podatnego na ataki. SVM dodaje nieliniowe granice decyzyjne, zwiększając odporność na niektóre ataki. XGBoost, jako model zespołowy, jest szeroko stosowany w praktyce, jest również modelem średnio dającym najlepsze rozwiązania - jeśli chodzi o dokładność predykcji - na danych benchmarkowych[3]. XGBoost może być zatem szczególnie atrakcyjnym algorytmem z punktów widzenia praktycznego zastosowania metod ataków - szkód jakie ataki mogą wywołać.

Regresja logistyczna (Logistic Regression, LR)

Regresja logistyczna (LR) to popularny model liniowy wykorzystywany do klasyfikacji binarnej. Modeluje prawdopodobieństwo, że dana obserwacja $\mathbf{x}$ należy do danej klasy y. Funkcja decyzyjna dla regresji logistycznej ma postać:

$$p(y = 1 \mid \mathbf{x}) = \frac{1}{1 + e^{-\mathbf{w}^T\mathbf{x}+b}}$$

gdzie $\mathbf{w}$ to wektor wag modelu, a b to parametr przesunięcia (bias). Reguła decyzyjna dla klasyfikacji jest następująca:

$$\hat{y} = \begin{cases} 1, & \text{jeśli } p(y = 1 \mid \mathbf{x}) \geqslant 0.5, \\ 0, & \text{w przeciwnym razie.} \end{cases}$$

Powód wyboru: Regresja logistyczna została wybrana ze względu na swoją prostotę oraz interpretowalność. Jako model liniowy jest często stosowana w praktycznych zastosowaniach, gdzie istotna jest przejrzystość decyzji. Jej liniowy charakter sprawia, że jest szczególnie podatna na ataki takie jak ZOO (Zeroth-Order Optimization) [18].

Charakterystyka:

- Typ modelu: model liniowy.

[3]Na podstawie wyników konkursów organizowanych przez Kaggle

- Złożoność obliczeniowa: $O(ndk)$, gdzie n to liczba próbek, d to liczba cech, a k to liczba iteracji w algorytmie gradientowym.
- Podatność na ataki: Wysoka (ze względu na liniową granicę decyzyjną).

Maszyna wektorów nośnych (Support Vector Machine, SVM)

Maszyna wektorów nośnych (SVM) to metoda uczenia maszynowego, która działa na zasadzie znajdowania optymalnej hiperpłaszczyzny oddzielającej od siebie punkty danych. Dla danych liniowo separowalnych funkcja decyzyjna $f(\mathbf{x})$ dla wejścia $\mathbf{x}$ ma postać:

$$f(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b,$$

gdzie $\mathbf{w}$ i b to odpowiednio wektor wag i przesunięcie. W przypadku danych nieliniowych SVM stosuje sztuczkę jądrową (kernel trick), w której dane są rzutowane do przestrzeni wyższej wymiarowości za pomocą funkcji $\phi(\mathbf{x})$. Wówczas funkcja decyzyjna przyjmuje postać:

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b,$$

gdzie $K(\mathbf{x}_i, \mathbf{x})$ to funkcja jądra (np. jądro RBF), a α_i to mnożniki Lagrange'a.

Powód wyboru: SVM zostało wybrane ze względu na swoją zdolność do modelowania złożonych granic decyzyjnych. Modele z jądrem nieliniowym (np. RBF) wykazują większą odporność na ataki. Jednakże ataki HopSkipJump są w stanie efektywnie przełamywać granice decyzyjne SVM [17].

Charakterystyka:

- Typ modelu: Model oparty na jądrach.
- Złożoność obliczeniowa: $O(n^2d)$ do $O(n^3)$, gdzie n to liczba próbek.
- Podatność na ataki: Średnia, zależna od zastosowanego jądra (większa odporność w przypadku jądra RBF).

XGBoost (Extreme Gradient Boosting)

XGBoost to metoda zespołowa, która tworzy serię drzew decyzyjnych, z których każde próbuje skorygować błędy swojego poprzednika. Funkcja predykcyjna $\hat{y}$ dla wejścia $\mathbf{x}$ ma postać:

$$\hat{y} = \sum_{k=1}^{K} f_k(\mathbf{x}),$$

gdzie $f_k(\mathbf{x})$ to predykcja k-tego drzewa, a K to liczba drzew. XGBoost minimalizuje funkcję straty L oraz dodaje człon regularyzacyjny $\Omega(f)$:

$$\mathcal{L}(\Theta) = \sum_{i=1}^{n} L(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k).$$

Powód wyboru: XGBoost jest jednym z najczęściej używanych algorytmów w praktyce, szczególnie w dziedzinach takich jak finanse i wykrywanie oszustw. Choć metody zespołowe są bardziej odporne na ataki, XGBoost wciąż może być celem ataków, takich jak PermuteAttack [32].

Charakterystyka:

- Typ modelu: Model zespołowy (drzewa decyzyjne).
- Złożoność obliczeniowa: $O(ndK)$, gdzie n to liczba próbek, d to liczba cech, a K to liczba drzew.
- Podatność na ataki: Średnia. Ataki na dane tabelaryczne, takie jak PermuteAttack, mogą zmniejszyć dokładność modelu.

Sztuczne sieci neuronowe

W drugiej fazie eksperymentów wykorzystano model sztucznej sieci neuronowej, składający się z następujących warstw:

- Warstwa wejściowa (input): warstwa liniowa przekształcająca dane wejściowe o wymiarze $n_features$ do przestrzeni 32-wymiarowej.
- Warstwa aktywacji (relu): nieliniowa funkcja aktywacji ReLU
- Warstwa wyjściowa (output): warstwa liniowa przekształcająca 32-wymiarową reprezentację do przestrzeni $n_class - wymiarowej$, odpowiadającej liczbie klas w zadaniu klasyfikacji.

Model został zoptymalizowany przy użyciu:

- Funkcji straty: entropii krzyżowej, standardowej funkcji straty w zadaniach klasyfikacji wieloklasowej.

- Optymalizatora: Adam ze współczynnikiem uczenia 0.01

Charakterystyka:

- Typ modelu: Płytka, w pełni połączona sieć neuronowa typu feed-forward (MLP - Multi-Layer Perceptron) z pojedynczą warstwą ukrytą. Model można opisać jako:
  - warstwa wejściowa: n_features → 32 neurony;
  - warstwa ukryta z aktywacją ReLU;
  - warstwa wyjściowa: 32 → n_class neuronów.
- Złożoność obliczeniowa: Relatywnie niska w porównaniu do modeli głębokiego uczenia.
  - Liczba parametrów: (n_features × 32) + 32 + (32 × n_class) + n_class.
  - Złożoność czasowa inferencji: O(n_features × 32 + 32 × n_class), czyli liniowa względem rozmiaru wejścia [33].
  - Złożoność pamięciowa: O(n_features × 32 + 32 × n_class).
- Podatność na ataki: Przedstawiony model może być szczególnie wrażliwy na niektóre typy ataków:
  - Ataki gradientowe (np. FGSM, PGD) [27, 52] - ze względu na prostą architekturę, gradient jest łatwy do obliczenia i manipulacji.
  - Ataki black-box z wykorzystaniem transferu ataku - mała liczba parametrów może prowadzić do większej podatności na ten typ ataków.
  - Ataki na cechy kategoryczne - przez liniowość pierwszej warstwy, perturbacje w zakodowanych cechach kategorycznych mogą mieć bezpośredni wpływ na predykcję [6] [61].

Prostota modelu ma pewne konsekwencje. Brak warstw regularyzacyjnych (np. dropout) zwiększa ryzyko nadmiernego dopasowania do próbek adwersaryjnych [72], a pojedyncza warstwa ukryta może być niewystarczająca do modelowania złożonych nieliniowych zależności, co może być wykorzystywane w atakach [55].

Wykorzystaniem sieci z jedną warstwą ukrytą jest kompromisem pomiędzy zdolnością modelu do reprezentowania złożonych relacji nieliniowych a jego interpretowalnością. Zgodnie z twierdzeniem o uniwersalnej aproksymacji, nawet sieć z pojedynczą

warstwą ukrytą jest w stanie aproksymować dowolną funkcję ciągłą na zwartym zbiorze [36]. Prostota architektury umożliwia śledzenie zmian w wewnętrznych reprezentacjach modelu, co jest trudniejsze w przypadku głębszych sieci, gdzie transformacje są bardziej złożone i mniej przejrzyste [27]. Modele o większej złożoności, takie jak głębokie sieci neuronowe, komplikują analizę i mogą maskować kluczowe zależności, co zostało potwierdzone w pracach dotyczących interpretowalności modeli [67].

Dane tabelaryczne charakteryzują się zwykle stosunkowo niewielką licznością próbek w stosunku do wymiarowości przestrzeni cech [70]. W przypadku głębokich sieci istnieje ryzyko nadmiernego dopasowania modelu do zbioru treningowego, co zmniejsza jego zdolność do generalizacji. Obecność warstwy ukrytej o wymiarze 32 pozwala modelowi na efektywne przetwarzanie zarówno cech numerycznych, jak i zakodowanych cech kategorycznych, jednocześnie ograniczając liczbę parametrów i unikając nadmiernej złożoności, która mogłaby prowadzić do przeuczenia [11] [72].

Wybór architektury był też podyktowany czynnikami praktycznymi. W przypadku danych tabelarycznych, gdzie relacje między cechami są zazwyczaj mniej złożone niż w przypadku danych sekwencyjnych czy obrazowych, płytka architektura sieci neuronowej często okazuje się wystarczająca do osiągnięcia zadowalającej skuteczności [71]. Prosta architektura sieci z jedną warstwą ukrytą znacząco redukuje czas potrzebny na proces uczenia modelu. Liczba parametrów w modelu rośnie liniowo wraz z liczbą neuronów, co umożliwia szybsze trenowanie i wielokrotne powtarzanie eksperymentów - możliwe jest przeprowadzanie licznych iteracji eksperymentów w różnych warunkach zakłóceń.

Bazując na powyższych, przyjęto prezentowany model jako dobrego kandydata do badań nad atakami adwersaryjnymi. Jego prostota ułatwia analizę mechanizmów ataku, ale jednocześnie zachowuje on podstawowe cechy typowe dla głębszych architektur neuronowych. Przedstawiony model reprezentuje standardową architekturę często używaną jako model odniesienia w badaniach nad bezpieczeństwem modeli uczenia maszynowego [14].

4.1.3 Metody ataku

Do celów eksperymentów wybrano sześć metod ataków. Trzy pierwsze metody (ZOO, HopSkipJump i PermuteAttack) wykorzystywane w podstawowym eksperymencie zakładają:

- brak dostępu do architektury modelu atakowanego oraz informacji o jego parametrach;
- brak dostępu do gradientów funkcji straty;
- możliwość wielokrotnego zapytania modelu;
- dostęp do etykiet decyzyjnych (hard labels).

W drugiej części eksperymentu zastosowano metody adwersaryjne dedykowane dla modeli opartych o sztuczne sieci neuronowe. Były to ataki: BIM, PGD i FGM. Metody te zakładają pełną wiedzę o architekturze modelu oraz dostęp do gradientów funkcji straty.

Metody ataków zostały zaimplementowane bazując na ich opisie literaturowym oraz kodzie dostępnym na platformie GitHub[4].

Założenia ogólne użyte przy opisie ataków:

- Model klasyfikacyjny $f(x) : \mathbb{R}^d \rightarrow \mathbb{R}^C$, gdzie d to wymiar danych wejściowych, a C liczba klas.
- Punkt wejściowy $x \in \mathbb{R}^d$, poprawnie klasyfikowany jako $y = \arg\max_c f_c(x)$, który ma być zakłócony w celu zmiany decyzji modelu na inną klasę.
- Funkcja straty $\mathcal{L}(x, y)$, która mierzy różnicę między predykcją modelu $f(x)$, a etykietą y.
- Zakłócenia $\Delta x = x' - x$, które są kontrolowane za pomocą normy L_p. $\|\Delta x\|_p \leqslant \epsilon$

Metoda ZOO

Atak **Zeroth Order Optimization** (ZOO) [18]to metoda generowania przykładów adwersaryjnych w środowisku *czarnej skrzynki*, gdzie model nie udostępnia informacji o gradientach funkcji straty. Technika ta wykorzystuje optymalizację zerowego rzędu, aproksymując gradienty za pomocą różnic skończonych. Umożliwia to modyfikowanie danych wejściowych w sposób ukierunkowany na zmianę decyzji modelu.

Założenia szczególne metody

Funkcja straty ma postać $\mathcal{L}(x, y)$, gdzie y to etykieta prawidłowa lub docelowa (w zależności od typu ataku adwersaryjnego: ukierunkowanego lub nieukierunkowanego).

[4]https://www.github.com/

Celem jest minimalizacja zakłócenia Δx, które powoduje zmianę klasyfikacji:

$$\min_{\Delta x} \mathcal{L}(x + \Delta x, y) \quad \text{z ograniczeniem:} \quad \|\Delta x\|_p \leqslant \epsilon.$$

Aproksymacja gradientu

Atak ZOO przybliża gradienty funkcji straty $\nabla_x \mathcal{L}(x, y)$ za pomocą różnic skończonych, co jest szczególnie istotne w środowisku *czarnej skrzynki* gdzie nie ma dostępu do wewnętrznych parametrów modelu. Gradient dla i-tej cechy oblicza się jako:

$$\frac{\partial \mathcal{L}(x, y)}{\partial x_i} \approx \frac{\mathcal{L}(x + he_i, y) - \mathcal{L}(x, y)}{h},$$

gdzie: h to mały krok perturbacji, e_i to wektor jednostkowy mający wartość 1 na i-tej pozycji.

Algorytm ataku

Atak ZOO realizuje iteracyjną optymalizację zakłócenia:

- Inicjalizacja: Punkt początkowy $x^{(0)}$ to oryginalne dane wejściowe x.
- Aktualizacja danych: W każdej iteracji obliczany jest przybliżony gradient, który służy do modyfikacji danych w kierunku maksymalizacji funkcji straty:

  $$x^{(t+1)} = x^{(t)} - \eta \cdot \frac{\nabla_x \mathcal{L}(x^{(t)}, y)}{\|\nabla_x \mathcal{L}(x^{(t)}, y)\|_p},$$

  gdzie η to krok optymalizacji.
- Projekcja: Jeśli $x^{(t+1)}$ przekracza ograniczenie $\|x^{(t+1)} - x\|_p \leqslant \epsilon$, to projektuje się je na dopuszczalny obszar:

  $$x^{(t+1)} = \text{Proj}_{\|\cdot\|_p \leqslant \epsilon}(x^{(t+1)}).$$

Kryteria zakończenia

Atak adwersaryjny kończy się, gdy:

- osiągnięto zmianę klasyfikacji $\arg\max_c f_c(x') \neq y$;
- przekroczono maksymalną liczbę iteracji; lub
- funkcja straty przestaje się zmieniać znacząco między iteracjami.

Zalety i ograniczenia

Atak ZOO działa w środowisku *czarnej skrzynki*, nie wymagając dostępu do gradientów ani parametrów modelu. Przy czym jest bardziej efektywny jeśli posiada dostęp do prawdopodobieństw klas (soft labels) - pozwala to na dokładniejszą aproksymację gradientów. W ataku tym można też elastycznie stosować różne normy L_p dla kontroli zakłóceń. Z drugiej strony charakteryzuje go wysoka złożoność obliczeniowa, szczególnie dla danych o dużych wymiarach ($O(T \cdot d)$, gdzie T to liczba iteracji, a d liczba cech). Ma też powolną zbieżność w przypadkach złożonych granic decyzyjnych.

Zastosowania

Atak ZOO znajduje zastosowanie w badaniach nad odpornością modeli w scenariuszach, gdzie dostęp do modelu jest ograniczony. Ze względu na uniwersalność metody, może być stosowany w różnych typach danych, od obrazów po dane tabelaryczne.

Metoda HopSkipJump

Atak **HopSkipJump** (HSJ) to iteracyjna metoda generowania przykładów adwersaryjnych, zaprojektowana do działania w środowisku *czarnej skrzynki* [17]. W odróżnieniu od ZOO, HSJ jest zoptymalizowana pod kątem minimalizacji liczby zapytań do modelu i zakłada dostęp jedynie do wynikowych etykiet (output labels). Mechanizm ataku łączy przeszukiwanie binarne oraz estymację granicy decyzyjnej.

Założenia szczególne

Celem jest znalezienie x', które spełnia:

$$\min_{x'} \|x' - x\|_p \quad \text{przy} \quad \arg\max_c f_c(x') \neq y.$$

Mechanizm działania

Atak HopSkipJump działa w trzech głównych etapach: inicjalizacja (Hop), przeszukiwanie binarne (Skip) oraz estymacja gradientu (Jump).

- Inicjalizacja (Hop): Początkowy punkt x'_0 jest znajdowany na przeciwnym boku granicy decyzyjnej $f(x') \neq y$. W praktyce stosuje się losowe wyszukiwanie lub algorytmy heurystyczne, aż znajdzie się punkt x'_0, dla którego klasyfikacja różni się od oryginalnej:

$$\arg\max_c f_c(x'_0) \neq y.$$

- Przeszukiwanie binarne (Skip): Znaleziony punkt x'_0 jest przybliżany do granicy decyzyjnej za pomocą przeszukiwania binarnego wzdłuż odcinka $[x, x'_0]$:

$$x'_b = \frac{x + x'_0}{2}.$$

- Jeśli $\arg\max_c f_c(x'_b) = y$, to $x \leftarrow x'_b$, - Jeśli $\arg\max_c f_c(x'_b) \neq y$, to $x'_0 \leftarrow x'_b$.

Proces powtarza się iteracyjnie, aż odległość między x, a x'_0 będzie mniejsza od ustalonej tolerancji ϵ.

- Estymacja gradientu (Jump): Granica decyzyjna jest dalej eksplorowana za pomocą estymacji gradientu w otoczeniu punktu x'_b. Gradient jest aproksymowany różnicami skończonymi:

$$\hat{\nabla} f(x) \approx \frac{f(x + \delta v) - f(x)}{\delta} v,$$

gdzie: δ to mała perturbacja, a v to losowy wektor jednostkowy.

Uzyskany kierunek jest wykorzystywany do aktualizacji:

$$x' \leftarrow x' - \eta \cdot \frac{\hat{\nabla} f(x')}{\|\hat{\nabla} f(x')\|_p},$$

gdzie η to krok optymalizacji.

Kryteria zakończenia
Atak kończy się, gdy:

- model zmieni klasyfikację na $\arg\max_c f_c(x') \neq y$;
- liczba iteracji osiągnie maksymalny próg; lub
- odległość $\|x' - x\|_p$ przestanie się znacząco zmieniać między iteracjami.

Zalety i ograniczenia

Atak HopSkipJump przede wszystkim minimalizuje liczbę zapytań do modelu dzięki przeszukiwaniu binarnemu. Niemniej - wymaga wielokrotnego testowania granicy decyzyjnej, co może być czasochłonne dla złożonych modeli. Atak wymaga liczby zapytań proporcjonalnej do wymiaru danych d na iterację dla estymacji gradientu oraz $O(\log K)$ zapytań dla przeszukiwania binarnego, gdzie K to odległość między x, a x'_0. Całkowita liczba zapytań wynosi:

$$O(T \cdot (d + \log K)),$$

gdzie: T to liczba iteracji.

Metoda PermuteAttack

PermuteAttack to metoda generowania przykładów adwersaryjnych zaprojektowana specjalnie dla danych tabelarycznych, które charakteryzują się obecnością zmiennych kategorycznych i dyskretnych. PermuteAttack działa w środowisku *czarnej skrzynki* i wykorzystuje algorytmy genetyczne do permutacji cech danych wejściowych, aby wygenerować skuteczne zakłócenia zmieniajce decyzję modelu.

Założenia szczególne

Celem jest minimalizacja liczby zmienionych cech $D_H(x, x')$ (odległość Hamminga), które prowadzą do zmiany decyzji modelu:

$$\min_{x'} D_H(x, x') \quad \text{przy} \ \arg\max_c f_c(x') \neq y.$$

Zakłócenia Δx muszą być zgodne z charakterem danych (np. wartości kategoryczne mogą być zmieniane tylko na inne dopuszczalne kategorie).

Mechanizm działania

Atak PermuteAttack wykorzystuje podejście oparte na algorytmach genetycznych, które iteracyjnie przeszukują przestrzeń możliwych permutacji cech.

- Inicjalizacja populacji
  - Tworzy się początkową populację $P^{(0)} = \{x'_1, x'_2, \ldots, x'_N\}$, gdzie każdy osobnik x'_i jest generowany przez losowe permutacje wybranych cech oryginalnego punktu x.
  - Permutacje są zgodne z ograniczeniami danych, np. dla zmiennych kategorycznych używane są tylko wartości z dozwolonego zbioru kategorii.
- Funkcja dopasowania Każdy osobnik w populacji x'_i jest oceniany za pomocą funkcji dopasowania:

  $$\mathcal{F}(x'_i) = \mathbb{1}(\arg\max_c f_c(x'_i) \neq y) - \lambda \cdot D_H(x, x'_i),$$

  gdzie:
  - Pierwszy składnik nagradza osobniki, które powodują zmianę decyzji modelu.
  - Drugi składnik penalizuje liczbę zmienionych cech.
  - λ to współczynnik równoważący wpływ obu składników.

- Selekcja i krzyżowanie Najlepsze osobniki z populacji $P^{(t)}$ są wybierane na podstawie funkcji dopasowania. Wybrane osobniki są poddawane operacji krzyżowania, która polega na wymianie wartości cech między osobnikami zgodnie z określonym prawdopodobieństwem.

- Mutacja Każdy osobnik podlega mutacji, polegającej na losowej permutacji jednej lub więcej cech, zachowując ograniczenia danych. Mutacja zwiększa różnorodność populacji i pozwala na eksplorację nowych obszarów przestrzeni możliwych przykładów przeciwstawnych.

- Aktualizacja populacji Powstaje nowa populacja $P^{(t+1)}$, w której najlepsi osobnicy są zachowywani (strategia elitarna), a pozostałe są generowane w wyniku krzyżowania i mutacji.

Kryteria zakończenia
Atak PermuteAttack kończy się, gdy wystąpi jeden z warunków:

- Znaleziono osobnika x', który spełnia $\arg\max_c f_c(x') \neq y$ przy minimalnym $D_H(x, x')$.

- Osiągnięto maksymalną liczbę iteracji.

- Populacja przestaje wykazywać poprawę funkcji dopasowania (konwergencja).

Zalety i ograniczenia
PermuteAttack był pierwszym atakiem w pełni obsługującym dane tabelaryczne z cechami kategorycznymi i dyskretnymi. Minimalizuje liczbę zmienionych cech, co zwiększa realizm zmodyfikowanych danych. Z drugiej strony generuje on wysoką liczba zapytań do modelu w przypadku dużych danych. Algorytmy genetyczne mogą być obliczeniowo kosztowne dla bardzo złożonych problemów. Dla populacji N, T iteracji i d cech: - Selekcja i ocena dopasowania: $O(N \cdot q)$, gdzie q to koszt jednego zapytania do modelu. - Krzyżowanie i mutacja: $O(N \cdot d)$. Łączna złożoność to:

$$O(T \cdot N \cdot (q + d)).$$

Metoda BIM

Atak **BIM** - "Basic Iterative Method" - jest pierwszym z wykorzystywanych w poniższych pracach atakiem działającym w paradygmacie *białej skrzynki* - zakłada

pełny dostęp do gradientów funkcji straty. Atak BIM wprowadza wielokrotne małe perturbacje danych wejściowych, aby sukcesywnie zbliżyć się do granicy decyzyjnej modelu, zachowując zakłócenia w granicach normy L_∞. [47] [14]

Założenia

Celem ataku jest znalezienie zmodyfikowanego przykładu x', który maksymalizuje stratę $\mathcal{L}(x', y)$, jednocześnie zachowując ograniczenie na zakłócenie:

$$x' = \arg\max_{x'} \mathcal{L}(x', y) \quad \text{przy } \|x' - x\|_\infty \leqslant \epsilon,$$

gdzie: ϵ to maksymalna dopuszczalna perturbacja dla każdej cechy.

Mechanizm działania

BIM działa w sposób iteracyjny, stopniowo zwiększając zakłócenie przy użyciu gradientów funkcji straty.

- *Aktualizacja iteracyjna* Zakłócenie w każdej iteracji jest obliczane na podstawie gradientu funkcji straty $\nabla_x \mathcal{L}(x, y)$, wprowadzając małe zmiany w kierunku maksymalizacji straty:

  $$x^{(t+1)} = x^{(t)} + \alpha \cdot \text{sign}(\nabla_x \mathcal{L}(x^{(t)}, y)),$$

  gdzie:

  - α to krok optymalizacji (mała wartość perturbacji w każdej iteracji),
  - $\text{sign}(\nabla_x \mathcal{L})$ wskazuje kierunek gradientu (zmiany w kierunku zwiększenia straty).

- *Projekcja na dopuszczalny obszar* Aby zakłócenie $x^{(t+1)}$ pozostało w granicach określonych przez ϵ, wynik każdej iteracji jest projektowany na sferę L_∞ wokół oryginalnego punktu x:

  $$x^{(t+1)} = \text{clip}(x^{(t+1)}, x - \epsilon, x + \epsilon).$$

  Zapewnia to, że maksymalna zmiana każdej cechy nie przekracza ϵ.

- *Inicjalizacja* Atak rozpoczyna się od oryginalnych danych wejściowych $x^{(0)} = x$.

Kryteria zakończenia

Atak kończy się, gdy:

- liczba iteracji osiągnie ustaloną wartość T; lub

- model zmieni klasyfikację: $\arg\max_c f_c(x') \neq y$.

Zalety i ograniczenia BIM jest rozwinięciem wcześniejszej metody FGSM, który wykonywał jednokrokową aktualizację zakłócenia. BIM wprowadza iteracyjne perturbacje o małym kroku, co pozwala na osiągnięcie wysokiej skuteczności ataku oraz tworzenie niewielkich i subtelnych zakłóceń – co się przekłada na jego niską wykrywalność. Zakłada on pełny dostęp do gradientów modelu, co oznacza możliwość jego stosowania jedynie w scenariuszach pełnego dostępu atakującego do modelu atakowanego. Złożoność obliczeniowa BIM jest proporcjonalna do liczby iteracji T i liczby cech d. Koszt każdej iteracji wynosi $O(d)$, co daje całkowitą złożoność $O(T \cdot d)$.

Metoda FGM

Fast Gradient Method (FGM) to jednokrokowy atak opracowany do generowania przykładów przeciwstawnych w środowisku *białej skrzynki*. Wykorzystuje gradient funkcji straty aby wprowadzić minimalne skuteczne zakłócenia w danych wejściowych. W odróżnieniu od bardziej iteracyjnych metod, takich jak BIM, w ataku FGM nacisk położony jest na efektywność obliczeniową i szybkość działania ataku. [27]

Założenia

Celem ataku jest wygenerowanie zmodyfikowanego przykładu x', który maksymalizuje stratę $\mathcal{L}(x', y)$, jednocześnie minimalizując wielkość zakłócenia:

$$x' = x + \epsilon \cdot \frac{\nabla_x \mathcal{L}(x, y)}{\|\nabla_x \mathcal{L}(x, y)\|_p},$$

gdzie:

- $\nabla_x \mathcal{L}(x, y)$ to gradient funkcji straty względem wejścia,
- ϵ to skalar kontrolujący wielkość zakłócenia,
- $\|\cdot\|_p$ to wybrana norma (najczęściej L_2 lub L_∞).

Mechanizm działania

- *Jednokrokowa aktualizacja* Zakłócenie Δx jest obliczane na podstawie gradientu funkcji straty:

  $$\Delta x = \epsilon \cdot \text{sign}(\nabla_x \mathcal{L}(x, y)),$$

  gdzie:

  - ϵ kontroluje maksymalny rozmiar zakłócenia,
  - sign$(\cdot)$ wskazuje kierunek gradientu (maksymalizujący stratę).

  Zmodyfikowany przykład adwersaryjny powstaje poprzez dodanie zakłócenia do danych wejściowych:

$$x' = x + \Delta x.$$

- *Projekcja na dopuszczalny obszar* Jeśli wymagane jest ograniczenie zakłócenia do określonej normy L_p, x' może zostać rzutowane na dopuszczalny obszar:

$$x' = \text{clip}(x + \Delta x, x - \epsilon, x + \epsilon),$$

  gdzie clip zapewnia, że x' pozostaje w zakresie wartości wejściowych.

Kryteria zakończenia

Ponieważ FGM jest metodą jednokrokową, atak kończy się natychmiast po wygenerowaniu x'. Decyzja modelu zostaje zmieniona, jeśli $\arg\max_c f_c(x') \neq y$.

Zalety i ograniczenia

FGW to bardzo szybka metoda generacji przykładów adwersaryjnych - wymaga jedynie pojedynczego obliczenia gradientu $\nabla_x \mathcal{L}(x, y)$. Całkowita złożoność wynosi $O(d)$, gdzie d to liczba cech danych wejściowych. Jest to też metoda prosta w implementacji. Zakłada jednakże pełny dostęp do gradientów modelu, co ogranicza zastosowanie w środowisku *czarnej skrzynki*. Dodatkowo - generowane zakłócenia są mniej precyzyjne w porównaniu do metod iteracyjnych, takich jak BIM.

Metoda PGD

Projected Gradient Descent (PGD) jest iteracyjną metodą generowania przykładów adwersaryjnych w warunkach pełnego dostępu do modelu atakowanego. PGD stanowi rozszerzenie metody BIM (Basic Iterative Method) i dodaje mechanizm projekcji, który pozwala na ograniczenie zakłóceń do wybranego obszaru w przestrzeni wejściowej, zgodnie z wybraną normą L_p. PGD jest uważany za jedno z najskuteczniejszych narzędzi w badaniu odporności modeli na ataki adwersaryjne. [52]

Założenia szczególne

Celem ataku PGD jest znalezienie x', które maksymalizuje stratę $\mathcal{L}(x', y)$, przy jednoczesnym ograniczeniu wielkości zakłócenia:

$$x' = \arg\max_{x'} \mathcal{L}(x', y) \quad \text{przy } \|x' - x\|_p \leqslant \epsilon,$$

gdzie ϵ to maksymalna dopuszczalna wielkość zakłócenia.

Mechanizm działania

PGD działa iteracyjnie, stopniowo zwiększając zakłócenia poprzez optymalizację funkcji straty.

- *Iteracyjna aktualizacja*

  Każda iteracja t generuje nowe zakłócenie Δx, które jest obliczane za pomocą gradientu funkcji straty względem danych wejściowych:

  $$x^{(t+1)} = x^{(t)} + \alpha \cdot \text{sign}(\nabla_x \mathcal{L}(x^{(t)}, y)),$$

  gdzie:

  - α to krok optymalizacji (wartość perturbacji w każdej iteracji),
  - $\text{sign}(\nabla_x \mathcal{L})$ wskazuje kierunek maksymalizacji straty.

- *Projekcja na dopuszczalny obszar* Po każdej aktualizacji przykład $x^{(t+1)}$ jest projektowany na sferę L_p wokół oryginalnego punktu x, aby zapewnić, że zakłócenie pozostaje w ustalonych granicach:

  $$x^{(t+1)} = \text{Proj}_{\|x'-x\|_p \leqslant \epsilon}(x^{(t+1)}).$$

  Projekcja ta gwarantuje, że maksymalne zakłócenie jest zgodne z normą L_p.

- *Inicjalizacja* Atak PGD zaczyna się od losowego punktu startowego $x^{(0)}$ w obszarze $\|x^{(0)} - x\|_p \leqslant \epsilon$. To losowe zainicjowanie zwiększa odporność ataku na mechanizmy obrony, takie jak wielokrotne uruchamianie.

Kryteria zakończenia

Atak PGD kończy się, gdy:

- liczba iteracji osiągnie ustaloną wartość T;
- odległość między kolejnymi iteracjami $\|x^{(t+1)} - x^{(t)}\|$ stanie się mniejsza niż ustalony próg; lub
- model zmieni klasyfikację: $\arg\max_c f_c(x^{(t)}) \neq y$.

Zalety i ograniczenia

Atak PGD jest bardzo skuteczny w generowaniu przykładów przeciwstawnych - mechanizm projekcji zapewnia, że zakłócenia pozostają w realistycznych granicach. Jest też odporny na mechanizmy obronne, takie jak randomizacja wejścia.

Kryterium	ZOO	HopSkipJump	PermuteAttack	FGM	BIM	PGD
Rodzaj środowiska	Czarna skrzynka: dostęp do wartości wyjściowych	Czarna skrzynka: dostęp do etykiet decyzji	Czarna skrzynka: dostęp do etykiet decyzji	Biała skrzynka: dostęp do gradientów	Biała skrzynka: dostęp do gradientów	Biała skrzynka: dostęp do gradientów
Typ danych	Dane ciągłe	Dane ciągłe	Dane numeryczne i kategoryczne	Dane ciągłe	Dane ciągłe	Dane ciągłe
Mechanizm optymalizacji	Aproksymacja gradientu różnicami skończonymi	Przeszukiwanie binarne + estymacja gradientu	Algorytm genetyczny oparty na permutacjach	Jednokrokowy gradient	Iteracyjne aktualizacje gradientu	Iteracyjne aktualizacje z losową inicjalizacją
Obsługa danych kategorycznych	Wymaga kodowania lub heurystyk	Ograniczona obsługa	Pełna obsługa	Brak w standardowej implementacji	Wymaga heurystyk lub ręcznych ograniczeń	Wymaga projekcji na dopuszczalne wartości
Uwzględnianie ograniczeń domenowych	Wymaga ręcznych modyfikacji	Zależne od implementacji	Naturalne uwzględnienie ograniczeń	Trudne do uwzględnienia	Możliwe z dodatkowymi ograniczeniami	Projekcja może uwzględniać ograniczenia domenowe
Złożoność obliczeniowa	O(T·d)	O(T·(d+logK))	O(T·N·(q+d))	O(d)	O(T·d)	O(T·d)
Liczba zapytań do modelu atakowanego	T·d	T·(d+logK)	T·N·q	Nie dotyczy: wymagany dostęp do gradientów	Nie dotyczy: wymagany dostęp do gradientów	Nie dotyczy: wymagany dostęp do gradientów
Skuteczność ataku	Wysoka: precyzyjne zakłócenia	Bardzo wysoka: precyzyjne i subtelne zakłócenia	Wysoka dla danych tabelarycznych	Średnia: efektywny na prostych modelach	Wysoka: bardziej skuteczny niż FGM	Bardzo wysoka: uznawany za standard
Realizm zakłóceń	Niski: zakłócenia mogą być nienaturalne	Średni: mniej zauważalne zmiany	Bardzo wysoki: zmiany zgodne z ograniczeniami	Niski: zakłócenia mogą być nienaturalne	Średni: bardziej subtelne niż FGM	Wysoki: subtelne dzięki projekcji

Rys. 4.1: Porównanie wybranych cech metod ataków rozważanych w rozprawie.

Charakteryzuje się wyższym kosztem obliczeniowym niż jednokrokowe ataki, takie jak FGSM. Złożoność PGD jest proporcjonalna do liczby iteracji T i liczby cech d. Każda iteracja wymaga obliczenia gradientu $\nabla_x \mathcal{L}(x, y)$, co daje całkowitą złożoność $O(T \cdot d)$. Ograniczeniem stosowania ataku PGD jest konieczność pełnego dostępu do gradientów modelu. PGD można postrzegać jako bardziej ogólną wersję BIM, w której wprowadzono mechanizm losowej inicjalizacji i projekcji na obszar L_p. W odróżnieniu od FGSM, PGD wykorzystuje iteracyjne podejście, co pozwala na generowanie bardziej precyzyjnych zakłóceń.

Porównanie wybranych metod ataków przedstawiono na rysunku 4.1.

4.1.4 Wykorzystywane oprogramowanie

Monitoring modeli uczenia maszynowego, tworzenie modeli zastępczych oraz atrybutów diagnostycznych wykonywane było przy użyciu oprogramowania BrightBox [41]. BrightBox opracowany został w QED Software, firma ta jest jednostką wdrażającą wyniki doktoratu. Oprogramowanie BrightBox zostało wykorzystane jako biblioteka programistyczna, umożliwiająca praktyczną implementację mechanizmów tworzenia modeli zastępczych oraz obliczania wartości atrybutów diagnostycznych.

Implementacja ataków na modele uczenia maszynowego została wykonana z użyciem pakietu programistycznego ART (Adversarial Robustness Toolkit), stworzonego na podstawie wyników grantów DARPA i dostępnego nieodpłatnie na platformie[5]GitHub. [58] Pakiet ART wykorzystywany był w wersji 1.18.

Dla ataków niewspieranych przez ART, korzystano przy implementacji udostępnianych przez autorów poszczególnych metod.

Eksperymenty prowadzone były z wykorzystaniem języka Pytnom w wersji 3.x (3.10 do 3.12).

4.2 Przebieg eksperymentów

4.2.1 Przygotowanie modeli uczenia maszynowego

W pierwszym etapie eksperymentów jako cele ataków wybrano trzy modele: Regresję Logistyczną (LR), Maszynę Wektorów Nośnych (SVM) oraz XGBoost (XGB). Modele te zostały wytrenowane i ocenione na zbiorze 22 zbiorów danych pochodzących z OpenML, obejmujących zadania klasyfikacji binarnej i wieloklasowej.

Przygotowanie danych

Z każdego zbioru danych wydzielono jego część mającą służyć do późniejszej diagnostyki modelu - co najmniej 100 przykładów. Dla każdego zbioru danych, pozostałą część podzielono na zbiór treningowy (70%) i zbiór testowy (30%). Zbiór treningowy był wykorzystywany do trenowania modeli, a zbiór testowy do oceny jakości wytrenowania.

W przypadku Regresji Logistycznej oraz SVM zastosowano normalizację metodą min-max, co pozwoliło przekształcić wartości cech do zakresu $[0, 1]$.

[5]https://github.com/Trusted-AI/adversarial-robustness-toolbox

Dane kategoryczne zostały zakodowane przy użyciu metody label encoding, przekształcając wartości kategoryczne w wartości numeryczne.

Brakujące wartości zostały uzupełnione medianą dla odpowiedniej cechy.

Proces trenowania modeli

Regresja Logistyczna (LR)

Funkcja celu: Regresja logistyczna minimalizuje funkcję straty krzyżowej entropii:

$$\mathcal{L}(\mathbf{w}) = -\frac{1}{n}\sum_{i=1}^{n}\left(y_i \log\left(p(y_i \mid \mathbf{x}_i)\right) + (1-y_i)\log\left(1-p(y_i \mid \mathbf{x}_i)\right)\right),$$

gdzie $p(y_i \mid \mathbf{x}_i) = \frac{1}{1+e^{-\mathbf{w}^T\mathbf{x}_i+b}}$, $\mathbf{w}$ to wektor wag, b to parametr przesunięcia (bias), a y_i to etykieta binarna.

Regularyzacja: Zastosowano regularyzację ℓ_2 (regularyzacja grzbietowa) w celu uniknięcia nadmiernego dopasowania modelu. Typowy zakres hiperparametru regularyzacji λ wynosi $\{0.001, 0.01, 0.1, 1\}$, co jest powszechnie stosowaną praktyką w literaturze.

Algorytm uczenia: Stochastyczny Spedek Gradientowy (SGD - Stochastic Gradient Descent) z wczesnym zatrzymaniem, gdy zmiana wartości straty jest mniejsza niż 10^{-4} przez 10 kolejnych iteracji.

Maszyna Wektorów Nośnych (SVM - Support Vector Machine)

Funkcja celu: SVM maksymalizuje margines między klasami i minimalizuje funkcję straty:

$$\mathcal{L}(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\max(0, 1-y_i(\mathbf{w}^T\mathbf{x}_i+b)),$$

gdzie C to parametr kary, który kontroluje kompromis pomiędzy maksymalizacją marginesu a minimalizacją błędu klasyfikacji.

Funkcja jądra: Wykorzystano jądro RBF (Radial Basis Function), które umożliwia modelowanie nieliniowych granic decyzyjnych:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right).$$

Standardowe hiperparametry:

- parametr C: $\{0.1, 1, 10, 100\}$,
- parametr γ: $\{0.001, 0.01, 0.1, 1\}$,

- jądro: RBF (Radial Basis Function).

Algorytm optymalizacji: Sequential Minimal Optimization (SMO) z kryterium zbieżności, gdy luka dwoistości jest mniejsza niż 10^{-3}.

XGBoost (XGB)

Funkcja celu: XGBoost minimalizuje skumulowaną funkcję straty i regularyzacji:

$$\mathcal{L}(\Theta) = \sum_{i=1}^{n} L(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k),$$

gdzie $L(y_i, \hat{y}_i)$ to strata entropii krzyżowej, a $\Omega(f_k)$ to kara regularyzacyjna dla k-tego drzewa.

Standardowe hiperparametry:

- współczynnik uczenia η: $\{0.01, 0.1, 0.2\}$,
- maksymalna głębokość drzew d: $\{3, 6, 10\}$,
- liczba drzew K: $\{50, 100, 200\}$,
- Subsample: 0.8.

Kryterium zatrzymania: Wczesne zatrzymanie po 10 kolejnych iteracjach bez poprawy wyniku na zbiorze walidacyjnym.

Wyniki uczenia modeli

Jakość trenowanych modeli oceniano za pomocą dwóch metryk: balanced accuracy (BA) oraz F1-score. Tabela 4.2 przedstawia średnią wartość balanced accuracy oraz F1-score dla każdego z typów modeli, uzyskaną na 22 zbiorach danych treningowych, a Tabela 4.3 wartości dla zbiorów testowych.

Model	**BA - średnie**	**BA - std. dev.**	**F1 - średnie**	**F1 - std. dev.**
RL	0.828	0.138	0.828	0.137
SVM	0.900	0.074	0.901	0.073
XGBoost	0.993	0.014	0.994	0.014

Tabela 4.2: Zbiorcze podsumowanie wyników trenowania modeli uczenia maszynowego dla zbiorów treningowych

Model	BA - średnie	BA - std. dev.	F1 - średnie	F1 - std. dev.
RL	0.790	0.136	0.790	0.135
SVM	0.855	0.097	0.857	0.095
XGBoost	0.852	0.098	0.856	0.099

Tabela 4.3: Zbiorcze podsumowanie wyników trenowania modeli uczenia maszynowego dla zbiorów testowych

W procesie trenowania XGBoost oaz SVM osiągnęły najwyższe uśrednione wyniki balanced accuracy i F1 score na zbiorze testowym na poziomie 0.85. Szczegółowe zestawienie jakości modeli, mierzone na zbiorze testowym, pokazane jest w Tabeli 4.4. Dla każdego zbioru danych co najmniej jeden z modeli osiągnął wystarczająco dobrą jakość klasyfikacji, a tylko dla dwóch przypadków jakość klasyfikacji odbiega znacząco od użytecznej. Zbiór modeli uznano za wystarczająco dobry z punktu widzenia późniejszych doświadczeń mających na celu ich zaatakowanie, przez co obniżenie w sposób zauważalny parametrów ich działania.

4.2.2 Przeprowadzenie ataków na wytrenowane modele ML

Dla każdej z kombinacji jednego z 22 zbiorów danych oraz jednego z trzech zaimplementowanych metod ataków, przeprowadzono każdy z trzech badanych ataków - w sumie przeprowadzono 198 ataków adwersaryjnych. Ataki zostały wykonane poprzez wprowadzenie celowych zaburzeń do fragmentów oryginalnych zbiorów danych, wcześniej wyodrębnionych jako część diagnostyczna (co najmniej 100 przykładów).

Ataki przeprowadzono z wykorzystaniem biblioteki ART [58] oraz repozytorium Permute Attack [32]. W atakach użyto standardowych parametrów skali ataku ϵ.

Podsumowanie wyników ataków przedstawiono na rysunku 4.2.

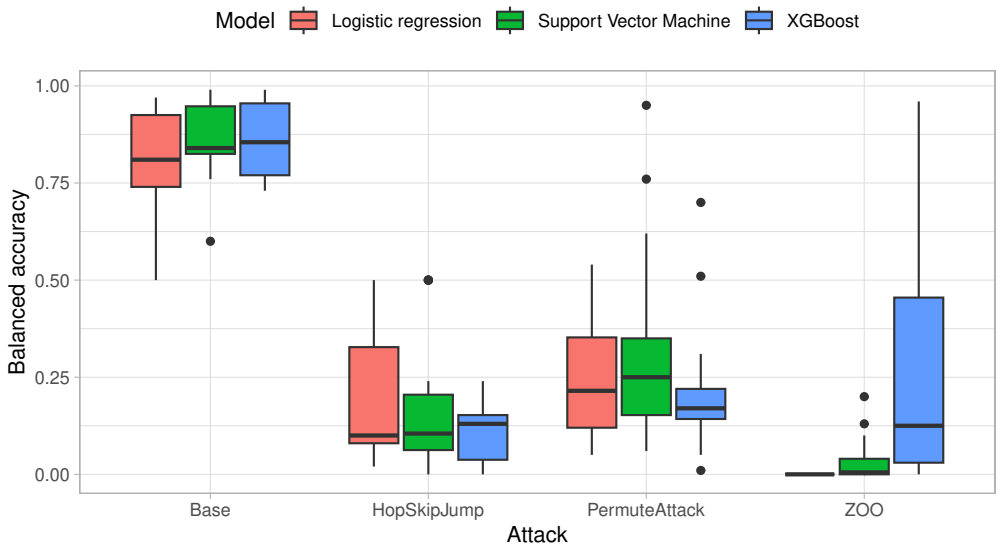
Na rysunku, na osi X widoczne są nazwy poszczególnych ataków oraz wyniki modelu niezaatakowanego (*Base*).

Przeprowadzone ataki adwersaryjne obniżają skuteczność modeli uczenia maszynowego, co widoczne jest w spadku wartości balanced accuracy. Mediana balanced accuracy dla modeli bazowych przekracza 0.8, jednak ataki takie jak HopSkipJump (HSJ) i Zeroth Order Optimization (ZOO) obniżają te wartości do poziomu odpowiednio 0.1 i bliskiego 0, co oznacza niemal całkowitą utratę zdolności klasyfikacyjnej. Atak PermuteAttack jest mniej skuteczny, a mediana balanced accuracy spada do około 0.2. Jest to zgodne z oczekiwaniami względem tego ataku - PermuteAttack

Model	RL	SVM	XGBoost
Bioresponse	0.742	0.772	0.784
churn	0.578	0.763	0.871
cmc	0.805	0.843	0.762
cnae-9	0.952	0.927	0.935
dna	0.938	0.949	0.957
har	0.971	0.970	0.992
madelon	0.576	0.597	0.730
mfeat-factors	0.805	0.844	0.761
mfeat-fourier	0.809	0.820	0.804
mfeat-karhunen	0.805	0.841	0.759
mfeat-zernike	0.805	0.843	0.771
nomao	0.927	0.942	0.958
optdigits	0.805	0.841	0.770
pendigits	0.933	0.992	0.989
phoneme	0.670	0.796	0.837
qsar-biodeg	0.742	0.774	0.806
satimage	0.808	0.886	0.905
semeion	0.914	0.949	0.924
spambase	0.805	0.841	0.756
wall-robot-navigation	0.605	0.836	0.987
wdbc	0.963	0.988	0.974
wilt	0.500	0.956	0.889

Tabela 4.4: Jakość poszczególnych modeli klasyfikacyjnych mierzona miarą balanced accuracy.

polega na losowej permutacji wartości cech wejściowych, co oznacza, że cechy są mieszane w sposób losowy bez uwzględnienia kierunku, w którym zmiany te mogłyby przesunąć punkt względem granicy decyzyjnej modelu. W przeciwieństwie do pozostałych dwóch testowanych ataków, które używają gradientów (lub ich przybliżeń) w celu przesunięcie wektora wejściowego w kierunku granicy decyzyjnej, PermuteAttack nie optymalizuje kierunku modyfikacji cech. W efekcie wiele z permutowanych przykładów pozostaje po tej samej stronie granicy decyzyjnej, przez co model nadal klasyfikuje je poprawnie. Powoduje to mniejszą skuteczność tego ataku przy ata-



Rys. 4.2: Rozkład wartości balanced accuracy dla wszystkich atakowanych modeli i zbiorów danych.

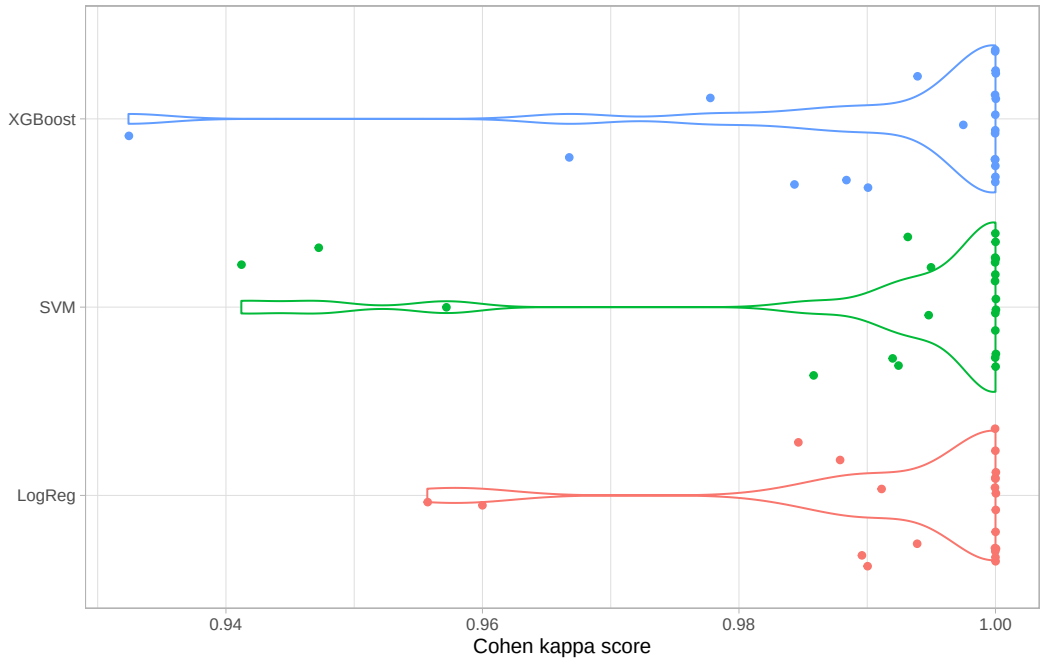
kowaniu modeli klasyfikujących dane numeryczne - dla danych kategorycznych ten atak jest atakiem pierwszego wyboru [32].

Wśród analizowanych modeli, XGBoost wykazuje najwyższą odporność, szczególnie wobec Permutation Attack, co wynika z zastosowania drzew decyzyjnych mniej wrażliwych na drobne zmiany wartości cech. SVM wykazuje umiarkowaną odporność, głównie dzięki wektorom nośnym definiującym granicę decyzyjną, natomiast RL jest najbardziej podatny, szczególnie na ataki ZOO i HSJ, które efektywnie przesuwają liniową granicę decyzyjną.

4.2.3 Analiza wartości atrybutów diagnostycznych

Mając przygotowane modele podlegające diagnozie oraz przygotowane przykłady adwersaryjne, można było przystąpić do wyliczenia wartości atrybutów diagnostycznych. W tym celu, dla każdego z diagnozowanych modeli przygotowany został model zastępczy, bazujący na 2500 reduktów aproksymacyjnych, dla których ($\epsilon = 0.05$). Weryfikacja właściwego stopnia podobieństwa modelu zastępczego do modelu diagnozowanego została przeprowadzona z wykorzystaniem miary kappa Cohena (κ), użytej do pomiaru podobieństwa decyzji podejmowanych przez model diagnozowany oraz model zastępczy. Przyjętym akceptowanym progiem podobieństwa była wartość κ większa od 0.9. Oznacza to ze modele zastępcze trenowano tak, aby zgodność de-

cyzji na zbiorach treningowych pomiędzy modelami diagnozowanymi, a zastępczymi spełniała warunek $\kappa \geqslant 0.9$. Warto zauważyć, że ze względu na sposób generowania modelu zastępczego i sposób weryfikacji zgodności decyzji, mamy gwarancję uzyskania modeli zastępczych spełniających przedstawione wymaganie zgodności poprzez wybranie odpowiedniej liczby reduktów i wartości ϵ. Jest tak dlatego, że model zastępczy powinien jedynie dobrze odzwierciedlać decyzje modelu diagnozowanego na danych treningowych.. Podsumowanie jakości stworzonych modeli zastępczych przedstawiono w 4.3.



Rys. 4.3: Rozkład wartości κ, porównujący jakość modeli zastępczych do modeli diagnozowanych.

Dla każdej pary [zbiór danych – atak] zostały wyliczone wartości atrybutów diagnostycznych. W tym celu najpierw dla każdej pary [zbiór danych – atak], badano zachowanie modelu zastępczego podczas analizy danych diagnostycznych, którymi było co najmniej 100 niezależnych obserwacji, wyodrębnionych przed fazą trenowania modeli ML podlegających diagnozie. Dla każdej obserwacji w diagnozowanym zbiorze danych obliczono i na tej podstawie obliczono wartości atrybutów diagnostycznych zdefiniowanych w Rozdziale 3.2 sąsiedztwo (ang. neighborhood).

Jak już wspominano w Rozdziale 3.2, sąsiedztwo to definiowane jest jako zbiór obserwacji pochodzących ze zbioru treningowego, które są podobne do danej obserwacji

z diagnozowanego zbioru danych. Zdefiniowane w ten sposób sąsiedztwo stanowi podstawę do obliczenia atrybutów diagnostycznych. Przypomnijmy, krótko znaczenie tych atrybutów:

- Spójność celu z przybliżeniami w sąsiedztwie (ang. target consistency with approximations in neighborhood) - miara określająca zgodność wartości celu diagnozowanej obserwacji z przybliżonymi wartościami predykcji uzyskanymi dla obserwacji znajdujących się w jej sąsiedztwie.

- Spójność predykcji z celami w sąsiedztwie (ang. prediction consistency with targets in the neighborhood) – miara określająca zgodność predykcji dla diagnozowanej obserwacji z wartościami celu obserwacji znajdujących się w jej sąsiedztwie.

- Spójność celu z celami w sąsiedztwie (ang. target consistency with targets in neighborhood) – miara opisująca zgodność wartości celu diagnozowanej obserwacji z wartościami celu obserwacji należących do jej sąsiedztwa.

- Niespójność celów i przybliżeń w sąsiedztwie (ang. targets and approximations inconsistency in neighborhood) – miara określająca stopień niespójności pomiędzy wartościami celu a przybliżeniami uzyskanymi dla obserwacji znajdujących się w sąsiedztwie diagnozowanej obserwacji.

- Różnorodność celów w sąsiedztwie (ang. targets diversity in nieghborhood) – miara opisująca zróżnicowanie wartości celu w sąsiedztwie diagnozowanej obserwacji w porównaniu z różnorodnością wartości celu obliczoną dla całego zbioru danych diagnozowanych.

- Różnorodność przybliżeń w sąsiedztwie (ang. approximations diversity in the neighborhood) – miara opisująca zróżnicowanie wartości przybliżonych predykcji w sąsiedztwie diagnozowanej obserwacji w odniesieniu do zróżnicowania przybliżeń obliczonych dla całego zbioru diagnozowanych danych.

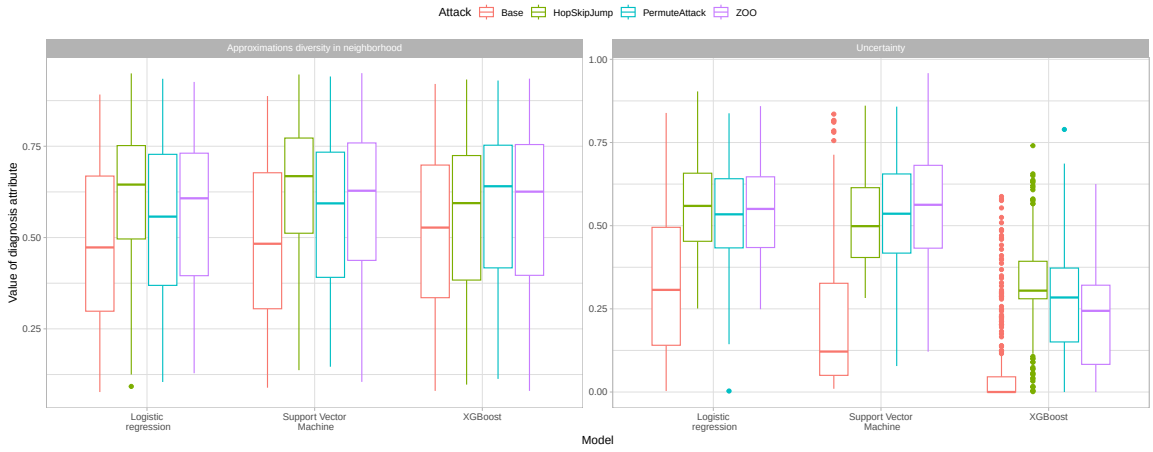
- Niepewność (ang. uncertainty)– miara niepewności predykcji, obliczana na podstawie rozkładu predykcji.

- Rozmiar sąsiedztwa (ang. neighborhood size) – liczba obserwacji znajdujących się w sąsiedztwie diagnozowanej obserwacji.

Jak juz wspominano są to podstawowe atrybuty diagnostyczne umożliwiające szczegółową charakterystykę otoczenia obserwacji w zbiorze danych w użytej metodzie.

Wartości atrybutów diagnostycznych obliczano dla każdego analizowanego zbioru danych i każdej z rozważanych metod ataku, oraz dla przypadku gdy dane diagnostycznie nie były zaburzone atakiem. Łącznie dało to 264 zbiorów wynikowych - 22 zbiory danych × 3 typy modeli (LS, SVM, XGBoost × 4 warianty ataków (brak ataku, ZOO, HopSkipJump, PermuteAttack).

Na rysunku 4.4 przedstawiono rozkład wartości dwóch atrybutów diagnostycznych (Różnorodności przybliżeń w sąsiedztwie oraz Niepewności) dla zbioru danych (*Spambase*). Wstępna analiza wykresu wskazuje na możliwość wystąpienia istotnych statystycznie różnic pomiędzy rozkładami atrybutów diagnostycznych przed i po ataku.



Rys. 4.4: Rozkład dwóch atrybutów diagnostycznych dla ataków przeprowadzanych z użyciem zbioru danych *Spambase*

W celu zweryfikowania hipotezy zerowej, zgodnie z którą nie występują istotne różnice pomiędzy rozkładem wszystkich mierzonych atrybutów diagnostycznych przed atakiem i po ataku, zastosowano test Kolmogorova-Smirnova (K-S). Test K-S jest nieparametrycznym testem statystycznym, który umożliwia ocenę zgodności dwóch rozkładów prawdopodobieństwa. Ponieważ jest to test nieparametryczny, jest on użyteczny w sytuacjach gdy nie można założyć określonego kształtu rozkładu danych. W prowadzonej analizie funkcje dystrybuanty empirycznej (ECDF) dla atrybutów diagnostycznych zostały wyznaczone oddzielnie dla danych sprzed ataku $F_{\text{before}}(x)$ oraz po ataku $F_{\text{after}}(x)$. Statystyka testowa Kolmogorova-Smirnova, zdefiniowana jako

maksymalna różnica pomiędzy tymi dwoma dystrybuantami, jest opisana równaniem:

$$D = \sup_{x \in \mathbb{R}} |F_{\text{before}}(x) - F_{\text{after}}(x)|,$$

gdzie: sup oznacza supremum, czyli największą wartość bezwzględnej różnicy pomiędzy dystrybuantami dla wszystkich wartości x z przestrzeni danych. Hipoteza zerowa H_0 zakłada że nie ma różnicy pomiędzy rozkładami atrybutów diagnostycznych przed i po ataku ($F_{\text{before}}(x) = F_{\text{after}}(x)$ dla każdego x). Hipoteza alternatywna H_1 zakłada, że rozkłady te różnią się w sposób istotny statystycznie. Decyzja o odrzuceniu hipotezy zerowej jest podejmowana na podstawie porównania wartości statystyki D z wartością krytyczną D_α, która jest wyznaczana na podstawie poziomu istotności α. W przypadku, gdy $D > D_\alpha$, hipoteza zerowa zostaje odrzucona.

W celu dodatkowej weryfikacji hipotezy o braku różnic w rozkładach atrybutów diagnostycznych przed i po ataku użyty został też test Wilcoxona. Test Wilcoxona jest nieparametrycznym testem rangowym, który umożliwia ocenę czy mediana różnic pomiędzy parami wartości atrybutów mierzonych przed atakiem i po ataku różni się istotnie od zera. W niniejszej analizie różnice pomiędzy parami wartości atrybutów diagnostycznych $d_i = d_{\text{after},i} - d_{\text{before},i}$ zostały uporządkowane według ich wartości bezwzględnej, a następnie przypisano im rangi. Testowa statystyka Wilcoxona W została obliczona jako suma rang przypisanych różnicom o dodatnich wartościach.

Hipoteza zerowa H_0 zakłada, że mediana różnic pomiędzy parami wartości atrybutów diagnostycznych przed atakiem i po ataku jest równa zero ($\text{Med}(d_i) = 0$). Hipoteza alternatywna H_1 głosi, że mediana tych różnic jest różna od zera ($\text{Med}(d_i) \neq 0$). Decyzja o odrzuceniu hipotezy zerowej jest podejmowana na podstawie porównania wartości statystyki W z wartością krytyczną W_α, która jest określona dla zadanego poziomu istotności α. W przypadku gdy W przekracza wartość krytyczną hipoteza zerowa zostaje odrzucona.

W obu przypadkach — zarówno dla testu K-S jak i testu Wilcoxona — przyjęto poziom istotności $\alpha = 0.05$. Taki poziom istotności oznacza, że ryzyko popełnienia błędu I rodzaju, czyli odrzucenia prawdziwej hipotezy zerowej, wynosi poniżej 0,05.

Dane - dla 198 różnych kombinacji 22 zbiorów danych, 3 ataków i 3 modeli klasyfikujących - zostały następnie zagregowane w celu uzyskania zbiorczego wskaźnika odsetka przypadków dla których hipoteza zerowa została odrzucona przy użyciu danego testu statystycznego.

W przypadku testu Kolmogorova-Smirnova, agregacja polegała na obliczeniu odsetka przypadków, w których maksymalna odległość pomiędzy dystrybuantami

empirycznymi atrybutów diagnostycznych przed atakiem i po ataku przekroczyła wartość krytyczną D_α. Wartość ta została określona na podstawie rozkładu statystyki testowej dla danego rozmiaru próby oraz poziomu istotności α.

Dla testu Wilcoxona, agregacja obejmowała obliczenie odsetka przypadków, w których hipoteza zerowa o równości median różnic pomiędzy parami wartości atrybutów przed atakiem i po ataku została odrzucona. Różnice pomiędzy parami były obliczane dla każdej obserwacji, a test Wilcoxona wykorzystywał sumy rang tych różnic do wyznaczenia statystyki W.

Ostateczne wyniki obu testów przedstawiono jako procent przypadków, w których hipoteza zerowa została odrzucona. W ten sposób łatwo dokonać oceny ogólnej skuteczności wykrywania różnic pomiędzy atrybutami diagnostycznymi przed i po ataku, oraz można zidentyfikować atrybuty szczególnie wrażliwe na wpływ ataku, co będzie następnie użyteczne przy projektowaniu docelowego klasyfikatora wykrywającego ataki adwersaryjne.

Wyniki testów statystycznych zostały przedstawione na trzech poziomach agregacji. Na każdym z nich zebrano, w danym ujęciu zbiorczyn, odsetek przypadków dla których hipoteza zerowa o braku różnic przed i po ataku została odrzucona z poziomem istotności $\alpha = 0.05$. Każdy poziom agregacji pozwala na ocenę efektywności w innym wymiarze, co ułatwia identyfikację kluczowych czynników wpływających na wykrywanie ataków.

- Poziom typu ataku (Tabela 4.5) – Na tym poziomie analizowano odsetek przypadków, w których hipoteza zerowa została odrzucona dla poszczególnych typów ataków (np. HopSkipJump, PermuteAttack, ZOO). Umożliwia to między innymi znalezienie ataków łatwiej bądź trudniej poddających się wykryciu.

- Poziom typu modelu(Tabela 4.6) – Na tym poziomie analizowano, jak skuteczność detekcji różnic w atrybutach diagnostycznych zależy od typu atakowanego modelu uczenia maszynowego poddawanego atakowi - LR, SVM, XGBoost. Dzięki temu można pokusić się o stwierdzenie podatności poszczególnych modeli na bycie atakowanymi w sposób trudny bądź łatwy do wykrycia.

- Poziom atrybutu diagnostycznego (Tabela 4.7) – Na tym poziomie można ocenić, które atrybuty diagnostyczne są najbardziej użyteczne w wykrywaniu ataków - które atrybuty diagnostyczne są najbardziej wrażliwe na zmiany wywołane atakami i mogą stanowić dane wejściowe do stworzenia klasyfikatora wykrywającego ataki.

Tabela 4.5: Procent wykrytych różnic pomiędzy wartościami atrybutów diagnostycznych - w dwóch testach statystycznych - agregacja po typie ataku

Typ ataku	K-S	Wilcox
HopSkipJump	88.28	94.91
PermuteAttack	79.36	94.63
ZOO	81.25	94.83

W przypadku ataku HopSkipJump test K-S odrzucił hipotezę zerową w 88% przypadków, podczas gdy test Wilcoxona odrzucił ją w 95% przypadków. Wykrywanie ataków Permute oraz ZOO okazało się nieco trudniejsze — skuteczność testu K-S wyniosła odpowiednio 79% i 81%. W przypadku testu Wilcoxona wartości te były zbliżone do 95%. Oznacza to, że choć skuteczność wykrywania różni się w zależności od rodzaju ataku, nawet w przypadku trudniejszych do wykrycia ataków, takich jak PermuteAttack i ZOO, które wprowadzają subtelniejsze lub mniej spójne zmiany, wciąż występuje istotna statystycznie różnica pomiędzy wartościami atrybutów diagnostycznych przed atakiem i po nim.

Dodatkowy wniosek wynikający z przeprowadzonej analizy jest taki, że test Wilcoxona jest bardziej czuły na subtelne zmiany mediany atrybutów diagnostycznych, nawet w przypadkach, gdy zmiany w całym rozkładzie są trudniejsze do wykrycia. Z kolei test K-S jest bardziej odpowiedni do wykrywania znaczących przesunięć w pełnym rozkładzie wartości atrybutów.

Tabela 4.6: Procent wykrytych różnic pomiędzy wartościami atrybutów diagnostycznych - w dwóch testach statystycznych - agregacja po typie modelu

Typ modelu	K-S	Wilcox
RL	84.47	92.59
SVM	85.61	94.24
XGBoost	78.52	97.75

Na poziomie typu modelu, z użyciem testu K-S wykryto 79% ataków przeprowadzonych na modelu XGBoost, prawie 84% ataków na modelu LR oraz 86% ataków na modelu SVM. W przypadku testu Wilcoxona skuteczność wykrywania ataków wyniosła 93% dla modelu LR, 94% dla modelu SVM oraz 98% dla modelu XGBoost. Nie stwierdzono znaczących różnic pomiędzy badanymi trzema modelami - dla każdego z nich wykrywalność ataków jest na podobnym poziomie.

Tabela 4.7: Procent wykrytych różnic pomiędzy wartościami atrybutów diagnostycznych - w dwóch testach statystycznych - agregacja po atrybucie diagnostycznym

Atrybut diagnostyczny	K-S	Wilcox
Różnorodność celów w sąsiedztwie	87.76	98.96
Rozmiar sąsiedztwa	72.45	96.35
Spójność predykcji z celami w sąsiedztwie	81.12	83.85
Spójność celu z przybliżeniami w sąsiedztwie	90.31	98.44
Spójność celu z celami w sąsiedztwie	87.24	95.83
Niespójność celów i przybliżeń w sąsiedztwie	60.71	89.58
Różnorodność celów w sąsiedztwie	88.27	96.35
Niepewność	95.41	98.95

Zgodnie z wynikami testu K-S, najwyższa skuteczność wykrywania różnic w rozkładach atrybutów diagnostycznych została osiągnięta dla trzech atrybutów. **Niepewność** uzyskała najwyższy wskaźnik wykrywania, na poziomie 95%, co wskazuje, że rozkład tego atrybutu ulega wyraźnej zmianie pod wpływem ataku. Drugim najskuteczniejszym atrybutem była **spójność celu z przybliżeniami w sąsiedztwie**, dla którego odsetek wykrytych zmian wyniósł 90%. Na trzecim miejscu - **Różnorodność celów w sąsiedztwie** z wynikiem 88%.

Wyniki testu Wilcoxona były zbliżone. **Niepewność** osiągnęła skuteczność na poziomie 98,95%, co stanowi najwyższy wynik spośród wszystkich atrybutów. **Różnorodność przybliżeń w sąsiedztwie zajęła** drugie miejsce z wynikiem 98,96% - świadczy to o dużej czułości tego atrybutu na zmiany wywołane atakami. Na trzecim miejscu znalazła się **Spójność celu z przybliżeniami w sąsiedztwie**, dla której test Wilcoxona wykazał istotne zmiany w 98,44% przypadków.

Powyższe wyniki można interpretować w następujący sposób:

- Niepewność jako kluczowy atrybut: Wysokie wskaźniki wykrywania (95% w teście K-S i 98,95% w teście Wilcoxona) sugerują, że niepewność jest jednym z najbardziej czułych atrybutów diagnostycznych do wykrywania ataków adwersaryjnych. W metodach opartych na uczeniu maszynowym wzrost niepewności predykcji jest często związany z przypadkami trudnymi do sklasyfikowania, co może być bezpośrednią konsekwencją wprowadzenia w ramach ataku niewielkich, celowych modyfikacji.

- Spójność celu z przybliżeniami: Wysoka skuteczność wykrywania zmian zarówno w teście K-S (90%) jak i Wilcoxona (98,44%) wskazuje, że ataki wpływają na spójność predykcji uzyskiwanych z przybliżeń. Ta miara odzwierciedla stabilność modelu w przewidywaniu celów na podstawie sąsiedztwa danych. Znaczące zmiany tej spójności po ataku mogą świadczyć o naruszeniu "lokalnej struktury" danych, co jest typowym efektem ataków adwersaryjnych.

- Różnorodność celów i różnorodność przybliżeń w sąsiedztwie: Różnorodność celów w sąsiedztwie wykazała wysoką skuteczność w teście K-S (88%), podczas gdy różnorodność przybliżeń w sąsiedztwie była lepiej wykrywana przez test Wilcoxona (98,96%). Oznacza to, że ataki adwersaryjne prowadzą do zmian w zróżnicowaniu lokalnych predykcji w sąsiedztwie, co może wynikać z celowego zakłócenia równowagi w danych przez atakującego. Te zmiany mogą obejmować zarówno zmiany w wartościach predykcji, jak i ich wzrost lub spadek w stosunku do wartości otaczających obserwacji.

Pomimo największej istotności powyżej wymienionych atrybutów diagnostycznych, stworzenie klasyfikatora wykrywającego ataki oraz jego późniejsze działanie oparto o pełen zestaw ośmiu atrybutów diagnostycznych.

4.2.4 Stworzenie klasyfikatorów wykrywających i rozróżniających ataki na podstawie atrybutów diagnostycznych

Po wstępnym potwierdzeniu zasadności przyjętej ścieżki prac, przystąpiono do konstrukcji klasyfikatora przyjmującego na wejściu agregaty wartości wszystkich atrybutów diagnostycznych, a na wyjściu zwracającego decyzję binarną [atak - brak wykrytego ataku] dla danej próbki danych. Dodatkowo - sprawdzono możliwość stworzenia klasyfikatora który może rozróżniać typy ataków spośród znanych ataków.

Do konstrukcji klasyfikatora binarnego wykrywającego ataki wykorzystano zbiór zawierający 264 wektory danych wejściowych. Każdy wektor danych odpowiadał jednej z kombinacji 22 pierwotnych zbiorów danych tabelarycznych **x** 3 modele je klasyfikujące **x** 4 przypadki ataków (3 ataki i 1 zbiór niezaatakwoany), i zawierał, wyliczone dla części diagnostycznej zbioru danych: agregaty wartości atrybutów diagnostycznych (opisane w dalszej części tej sekcji), liczność zbioru diagnostycznego, liczbę klas decyzyjnych, balanced accuracy pierwotnego klasyfikatora przetwarzającego dane diagnostyczne. Zawierał też etykietę binarną - atak lub brak ataku.

Do próby stworzenia klasyfikatora rozróżniającego ataki wykorzystano ten sam zbiór danych, różniący się jedynie wartością etykiety ataku - zawierał cztery klasy decyzyjne (etykieta braku ataku oraz etykiety dla każdego z badanych ataków: HopSkipJump, ZOO, PermuteAttack).

Punktem startowym był zestaw ośmiu atrybutów diagnostycznych wyznaczonych dla każdej instancji w zbiorze danych diagnostycznych oraz dla każdego ataku. W celu przygotowania odpowiednich danych treningowych dla klasyfikatora dokonano agregacji wartości atrybutów w przekrojach zdefiniowanych przez zbiór danych, model oraz rodzaj ataku. Dla każdego atrybutu diagnostycznego obliczono zestaw statystyk opisowych, obejmujący: wartość średnią, minimum, maksimum, kwartyl dolny, medianę, kwartyl górny oraz maksymalny rozstęp. Do tak otrzymanych agregatów dołączono charakterystyki zbioru danych - liczbę obserwacji, liczbę klas w problemie klasyfikacyjnym oraz wartość balanced accuracy wyznaczoną na podstawie porównania wyników predykcji i wartości rzeczywistych dla danych diagnostycznych. Łącznie analizie poddano 59 cech, stanowiących wektor danych wejściowych do klasyfikatora wykrywającego ataki. Tabela 4.8 prezentuje pierwsze 15 wierszy oraz 7 kolumn zagregowanego zbioru danych, wykorzystanego do przygotowania klasyfikatora.

Tabela 4.8: Fragment zagregowanej tabeli zawierającej dane użyte do stworzenia klasyfikatorów wykrywających i rozróżniających ataki. Atak *org* oznacza dane bez ataku.

Zbiór danych	Model	Atak	BA	N_obs	Neigh_size _mean	Neigh_size _Q0	···
Bioresponse	lin	hsj	0.26	751.00	178.06	67.00	···
Bioresponse	lin	org	0.74	751.00	264.75	122.00	···
Bioresponse	lin	per	0.36	751.00	263.78	122.00	···
Bioresponse	lin	zoo	0.00	751.00	261.08	120.00	···
Bioresponse	svm	hsj	0.50	751.00	277.85	66.00	···
Bioresponse	svm	org	0.77	751.00	408.08	208.00	···
Bioresponse	svm	per	0.52	751.00	402.40	207.00	···
Bioresponse	svm	zoo	0.02	751.00	358.64	95.00	···
Bioresponse	xgb	hsj	0.22	751.00	178.99	77.00	···
Bioresponse	xgb	org	0.78	751.00	290.35	150.00	···
Bioresponse	xgb	per	0.31	751.00	285.49	151.00	···
Bioresponse	xgb	zoo	0.64	751.00	290.25	148.00	···
churn	lin	hsj	0.42	1000.00	9.57	2.00	···
churn	lin	org	0.58	1000.00	8.20	5.00	···
churn	lin	per	0.42	1000.00	5.49	4.00	···
...	...	...	...	...	...	...	...

Znaczenie nazw wybranych kolumn jest następujące: *Model* zawiera oznaczanie użytych modeli ML; *Atak* oznacza wybrany typ ataku lub dane bez ataku; *BA* oznacza balanced accuracy; *N_obs* oznacza liczbę punktów danych (liczność klasyfikowanego zbioru diagnostycznego); *Neigh_size_mean* oznacza średnią wielkość sąsiedztwa; *Neigh_size_Q*0 oznacza granicę dla pierwszego kwantylu. Pełna tabela zawiera w sumie 59 kolumn.

Wartości w kolumnie *Model* wskazują na:

- *lin*, model uzyskany za pomocą regresji logistycznej,
- *svm*, oznacza model uzyskany za pomocą metody SVM,
- *xgb*, oznacza model uzyskany za pomocą algorytmu XGBoost.

Ponadto, w kolumnie *Atak*, wartość *org* oznacza zbiór danych bez ataku, a wartości *hsj, per, zoo* wskazują odpowiednio na metody ataku HopSkipJump, PermuteAttack i ZOO.

Tabela 4.8 przedstawia dane w formie w jakiej zostały użyte zarówno do stworzenia klasyfikatora wykrywającego fakt ataku, jak i do próby stworzenia klasyfikatora rozróżniającego poszczególne ataki. Na potrzeby zagadnienia wykrywania ataku, kolumna *attack* oznaczająca typ ataku została zastąpiona zmienną binarną, przyjmującą wartość 0 dla przypadku braku ataku, oraz wartość 1 w przeciwnym przypadku.

W celu stworzenia wiarygodnych klasyfikatorów zaprojektowano zestaw scenariuszy eksperymentalnych, których charakterystyka została przedstawiona w Tabeli 4.9. Pierwsza kolumna tabeli zawiera nazwę scenariusza, druga zwięzły opis głównej idei stojącej za danym scenariuszem, natomiast trzecia kolumna określa warunki stosowalności scenariusza. Każdy scenariusz, z wyjątkiem ostatniego, został zrealizowany dla obu typów zadań — detekcji ataku oraz izolacji typu ataku. Ostatni scenariusz ma zastosowanie wyłącznie w przypadku detekcji, ponieważ celem jest nauczenie modelu rozpoznawania ataków, w tym również tych, których charakterystyki nie zostały uwzględnione w danych treningowych.

Tabela 4.9: Scenariusze testowe klasyfikatorów

Scenariusz	Opis	Zastosowanie
Walidacja krzyżowa	Dziesięciokrotna walidacja krzyżowa	wykrywanie, izolacja
Usunięcie jednego zbioru danych	Jeden ze zbiorów danych traktowany jest jako zbiór testowy - klasyfikator jest trenowany na pozostałych zbiorach	wykrywanie, izolacja
Usunięcie jednego modelu	Wszystkie przypadki analizowane przez jeden z typów modeli są traktowane jako dane testowe - klasyfikator jest trenowany na pozostałych przypadkach.	wykrywanie, izolacja
Usunięcie jednego ataku	Wszystkie przypadki użycia tego samego ataku są traktowane jako dane testowe - klasyfikator jest trenowany na pozostałych danych	wykrywanie

Dla każdego scenariusza wytrenowano modele klasyfikacyjne bazujące na metodzie lasów losowych (random forest) [12] oraz używając algorytmu XGBoost. Podczas trenowania dokonano optymalizacji hiperparametrów tych klasyfikatorów na niezależnych od przykładów testowych zbiorach przykładów.

Klasyfikator - wykrywanie ataku

W pierwszym eksperymencie postawiono za cel opracowanie i wytrenowanie klasyfikatorów binarnych dokonujących rozróżnienia między dwoma klasami: stanem normalnym (brak ataku) a wystąpieniem incydentu (dowolny typ ataku). To podstawowe rozróżnienie stanowi kluczowy element w procesie detekcji potencjalnych zagrożeń w systemie korzystającym z modelu ML. Modele random forest oraz XGBoost były trenowane zgodnie z przedstawionymi w Tabeli 4.9 scenariuszami. Szczegółowe wyniki miary blanced accuraty, uzyskane dla wytrenowanych modeli zostały zaprezentowane w Tabeli 4.10.

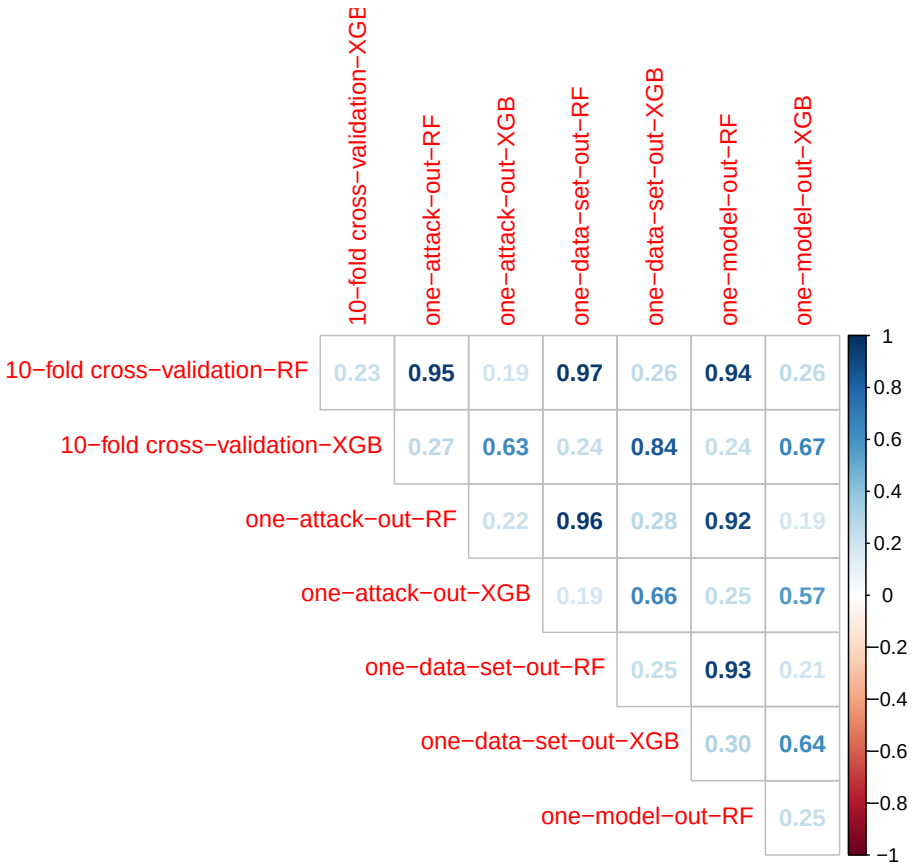
Tabela 4.10: Balanced accuracy oraz jej odchylenie standardowe dla klasyfikatorów wykrywających ataki.

Nazwa scenariusza	**Random Forest**	**XGBoost**
10-krotna walidacja krzyżowa	0.9107 (0.1050)	0.9557 (0.0679)
Usunięcie jednego ataku	0.9495 (0.0437)	0.9596 (0.0350)
Usunięcie jednego zbioru danych	0.9015 (0.1599)	0.9520 (0.1167)
Usunięcie jednego modelu	0.9364 (0.0441)	0.9164 (0.0621)

Wniosek - dla obydwu użytych modeli uczenia maszynowego możliwe jest zidentyfikowanie ataku bazując na wykorzystaniu atrybutów diagnostycznych. Najwyższą zbalansowaną dokładność uzyskano w scenariuszu one-attack-out dla obu klasyfikatorów, co wskazuje na zdolność rozpoznawania faktu wystąpienia ataku nawet w przypadku ataków niewystępujących wcześniej w zbiorze treningowym (ale oczywiście przeprowadzonych na inne zbiory danych niż zbiór testowy). Najniższą skuteczność zaobserwowano w scenariuszu one-data-set-out dla klasyfikatora typu random forest — dla nowego zbioru danych identyfikacja ataku okazała się trudniejsza niż np. dla nowego typu modelu. Z punktu widzenia praktycznej implementacji wskazuje to na lepszą adaptowalność klasyfikatorów opartych o XGBoost.

Dla każdego scenariusza i klasyfikatora wyznaczono istotność cech i utworzono ich ranking — wyższa pozycja oznacza większą istotność. Obliczono współczynniki korelacji między tymi rangami dla różnych scenariuszy. Wyniki wskazują na wysoką korelację porządku cech w obrębie grup klasyfikatorów. Przykładowo, współczynnik korelacji między rangami istotności cech w scenariuszach 10-krotnej walidacji krzyżowej i one-data-set-out dla klasyfikatora random forest wynosi 0.97. Współczynnik korelacji pomiędzy klasyfikatorami opartymi o XGBoost a random forest jest relatywnie niski — waha się od 0.19 do 0.30 (4.5), co wskazuje, że dla analizowanych klasyfikatorów różne zmienne mają kluczowe znaczenie w identyfikacji ataków. Rysunek 4.6 przedstawia 20 najistotniejszych zmiennych dla każdego klasyfikatora.

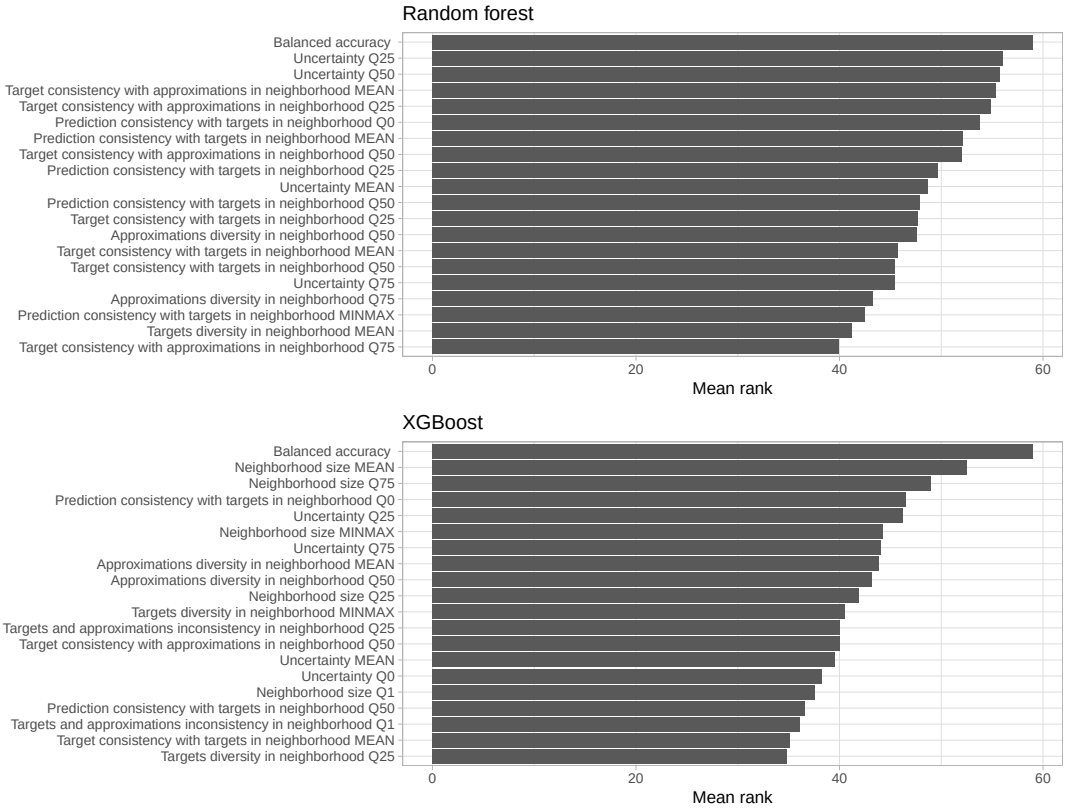
W przeprowadzonej analizie istotności atrybutów diagnostycznych dla obu typów klasyfikatorów kluczową rolę odgrywa balanced accuracy wyznaczona na zbiorze testowym. W przypadku klasyfikatora random forest, na drugim i trzecim miejscu znajdują się cechy powiązane z atrybutem diagnostycznym *Niepewność*, podczas gdy dla XGBoost pozycje te zajmują cechy powiązane z *Rozmiarem sąsiedztwa*.



Rys. 4.5: Macierz korelacji pomiędzy jakością klasyfikatorów (random forest i XGBoost) w różnych scenariuszach treningowych. Jakość klasyfikatora mierzono jako wartość balanced accuracy na danych testowych.

Badanie częstości występowania cech w pierwszej dwudziestce najistotniejszych wskazuje na najczęstsze występowanie cech opartych o atrybut diagnostyczny *Spójność predykcji z celami w sąsiedztwie* w przypadku random forest oraz cech opartych o*Rozmiar sąsiedztwa* dla XGBoost. Zaobserwowane różnice w hierarchii istotności atrybutów diagnostycznych świadczą o tym, że każda z wprowadzonych charakterystyk wnosi istotną informację do procesu detekcji ataków, przy czym modele w różny sposób wykorzystują dostępne atrybuty diagnostyczne.

Uzyskane wyniki mają istotne implikacje praktyczne dla konstrukcji klasyfikatorów wykrywających ataki. Obserwowana dominacja miary rozmiaru sąsiedztwa w klasyfikatorze XGBoost sugeruje, że model ten w znacznym stopniu opiera się na analizie lokalnej gęstości danych, co może wskazywać na jego skuteczność w identyfikacji przykładów adwersaryjnych poprzez wykrywanie regionów o nietypowej



Rys. 4.6: Średnia ranga istotności atrybutów diagnostycznych dla obydwu badanych typów klasyfikatorów.

reprezentacji w przestrzeni cech. Z kolei wysoka istotność zgodności predykcji z wartościami rzeczywistymi w sąsiedztwie dla klasyfikatora random forest wskazuje na wykorzystanie przez ten model spójności lokalnej struktury danych.

Zaobserwowane prawidłowości sugerują również, że efektywny system detekcji ataków powinien uwzględniać zarówno charakterystyki lokalne (Rozmiar sąsiedztwa), jak i miary spójności predykcji w zdefiniowanych obszarach przestrzeni cech.

Klasyfikator - wykrywanie typu ataku

Po pozytywnym zakończeniu eksperymentu związanego z wykrywaniem ataku, podjęto próbę stworzenia skutecznych klasyfikatorów rozróżniających ataki. Wytrenowano klasyfikatory random forest oraz XGBoost, ustalając czterowartościową zmienną decyzyjną: brak ataku, atak HopSkipJump, atak PermuteAttack oraz atak ZOO. Wyniki uzyskane dla poszczególnych scenariuszy zostały przedstawione w Tabeli 4.11.

Tabela 4.11: Balanced accuracy oraz odchylenie standardowe dla każdego z rozważanych w pracy scenariuszy trenowania klasyfikatorów.

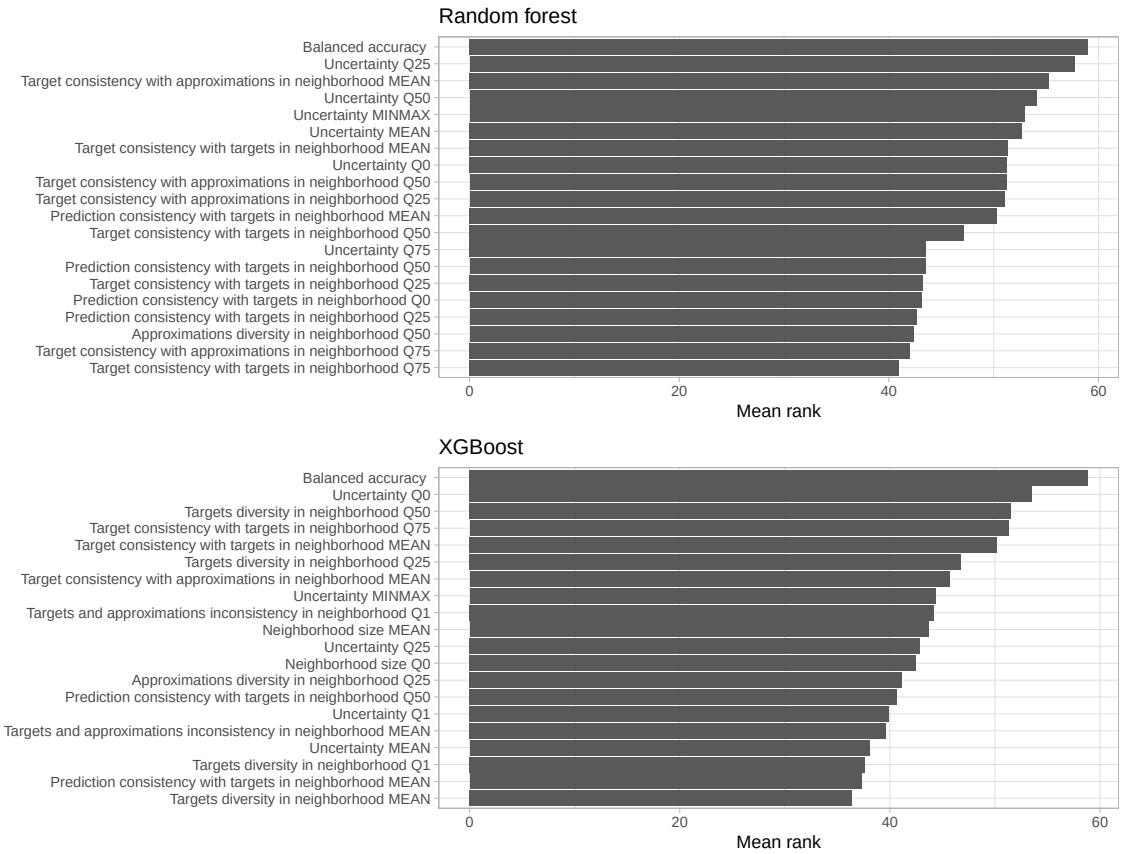
Nazwa scenariusza	**Random Forest**	**XGBoost**
10-krotna walidacja krzyżowa	0.6560 (0.1542)	0.7625 (0.0961)
Usunięcie jednego zbioru danych	0.7443 (0.2213)	0.8030 (0.1512)
Usunięcie jednego modelu	0.7102 (0.1476)	0.7216 (0.1379)

W kolejnym etapie eksperymentów przeprowadzono proces trenowania klasyfikatorów random forest oraz XGBoost w celu rozróżniania typów ataków. W tym zadaniu zmienna decyzyjna przyjmowała cztery klasy: brak ataku, atak HopSkipJump, atak PermuteAttack oraz atak ZOO. Wyniki uzyskane dla poszczególnych scenariuszy zostały zaprezentowane w Tabeli 4.11.

Analiza tabeli oraz jej porównanie z wynikami z poprzedniej części eksperymentu, wykazują iż izolacja ataków jest znacznie trudniejszym zadaniem niż ich wykrywanie.

Modele bazujące na XGBoost osiągają wyższe wartości balanced accuracy niż klasyfikatory random forest. Najwyższe wartości dokładności uzyskano w scenariuszu treningowym zakładającym usunięcie jednego zbioru danych, co sugeruje, że rozróżnianie typów ataków na nowym zbiorze danych jest łatwiejsze niż w przypadku konieczności rozpoznawania typów ataków dla nowego rodzaju modelu.

Podobnie jak w przypadku wykrywania ataków, analiza rankingu istotności cech wykazała, że ich kolejność jest silnie skorelowana w obrębie tego samego typu klasyfikatora. Współczynniki korelacji pomiędzy rankingami cech dla różnych scenariuszy realizowanych z użyciem tego samego klasyfikatora mieszczą się w przedziale od 0.67 do 0.99. Współczynniki korelacji pomiędzy rankingami cech dla klasyfikatorów random forest i XGBoost są wyższe niż te uzyskane w zadaniu wykrywania ataków i mieszczą się w zakresie od 0.41 do 0.51. Wyniki te sugerują, że choć klasyfikatory random forest i XGBoost kładą nacisk na różne cechy, to w zadaniu izolacji ataków ich podejście do identyfikacji istotnych atrybutów jest bardziej zbieżne niż w przypadku wykrywania ataków. Na Rysunku 4.7 przedstawiono średnie rangi 20 najważniejszych cech wykorzystywanych przez analizowane klasyfikatory.



Rys. 4.7: Średnia ranga istotności cech dla obydwu badanych typów klasyfikatorów.

Podobnie jak w przypadku wykrywania ataków, najważniejszą cechą wypływającą na skuteczność izolacji ataków jest balanced accuracy. Na kolejnych miejscach w rankingach istotności znajdują się zmienne powiązane z miarami niepewności. Wśród 20 najważniejszych cech wykorzystywanych przez analizowane klasyfikatory, random forest opiera swoje decyzje na statystykach obliczonych na podstawie pięciu atrybutów diagnostycznych, podczas gdy XGBoost uwzględnia osiem takich atrybutów. Sugeruje to, że XGBoost może lepiej wykorzystywać informację zawartą w różnych atrybutach diagnostycznych.

4.2.5 Weryfikacja skuteczności klasyfikatora na nowych atakach i dla nowych typów modeli uczenia maszynowego

Przeprowadzone eksperymenty potwierdziły skuteczność zaproponowanego podejścia w wykrywaniu ataków na klasyczne modele uczenia maszynowego - regresję logistyczną, SVM oraz XGBoost. Współczesne systemy w coraz większym stopniu opierają się na różnych architekturach sieci neuronowych, które ze względu na swoją złożoną architekturę i nieliniowy charakter mogą być szczególnie podatne na ataki adwersaryjne. Dlatego w kolejnym kroku postanowiono zbadać jak przedstawione podejście sprawdzi się przy wykrywaniu ataków na takie właśnie modele. Warto podkreślić, że pomimo rozwoju tzw. głębokich sieci neuronowych i ich szerokiemu zastosowaniu np. w klasyfikacji obrazów w przypadku analizy danych tabelarycznych nie opracowano do tej pory zbyt wielu architektur zapewniających wyniki predykcji średnio lepsze niż klasyczne metody analityczne rozważane w poprzednim podrozdziale.

Dotychczas badane ataki - ZOO, HopSkipJump oraz PermuteAttack reprezentują kategorię ataków *czarnej skrzynki*, które nie wymagają dostępu do struktury wewnętrznej modelu ani jego parametrów. Jednak w przypadku głębokich sieci neuronowych szczególnie istotne jest zbadanie odporności na ataki typu *biała skrzynka*, które wykorzystują pełną wiedzę o architekturze i parametrach modelu do generowania przykładów adwersaryjnych. Ta klasa ataków, choć trudniejsza do zrealizowania w praktyce, pozwala na ocenę najbardziej krytycznych podatności modeli głębokiego uczenia.

Jako ataki testowe wybrano trzy przykładowe i nowoczesne ataki - BIM, PGD oraz FGM, szerzej opisane w podrozdziale **4.1.3**.

Na początku, na tych samych co użyte w reszcie eksperymentów zbiorach danych, wytrenowano klasyfikatory bazujące na sieci neuronowej przeznaczonej do klasyfikacji danych tabelarycznych/ Struktura sieci opisana została w podrozdziale **4.1.2**. Struktura modelu została wygenerowana przy użyciu następującego kodu:

```
class TabDataModel(nn.Module):
    def __init__(self, n_features, n_class):
        super().__init__()
        self.input = nn.Linear(n_features, 32)
        self.relu = nn.ReLU()
        self.output = nn.Linear(32, n_class)
    def forward(self, x):
```

```
        return self.output(self.relu(self.input(x)))

model = TabDataModel(n_features, n_class)

loss_fn = nn.CrossEntropyLoss()
optimizer = optim.Adam(model.parameters(), lr=0.01)
```

Liczba epok uczenia zależała od konkretnego zbioru danych, ale zwykle satysfakcjonujące wyniki otrzymywano dla wartości 5000.

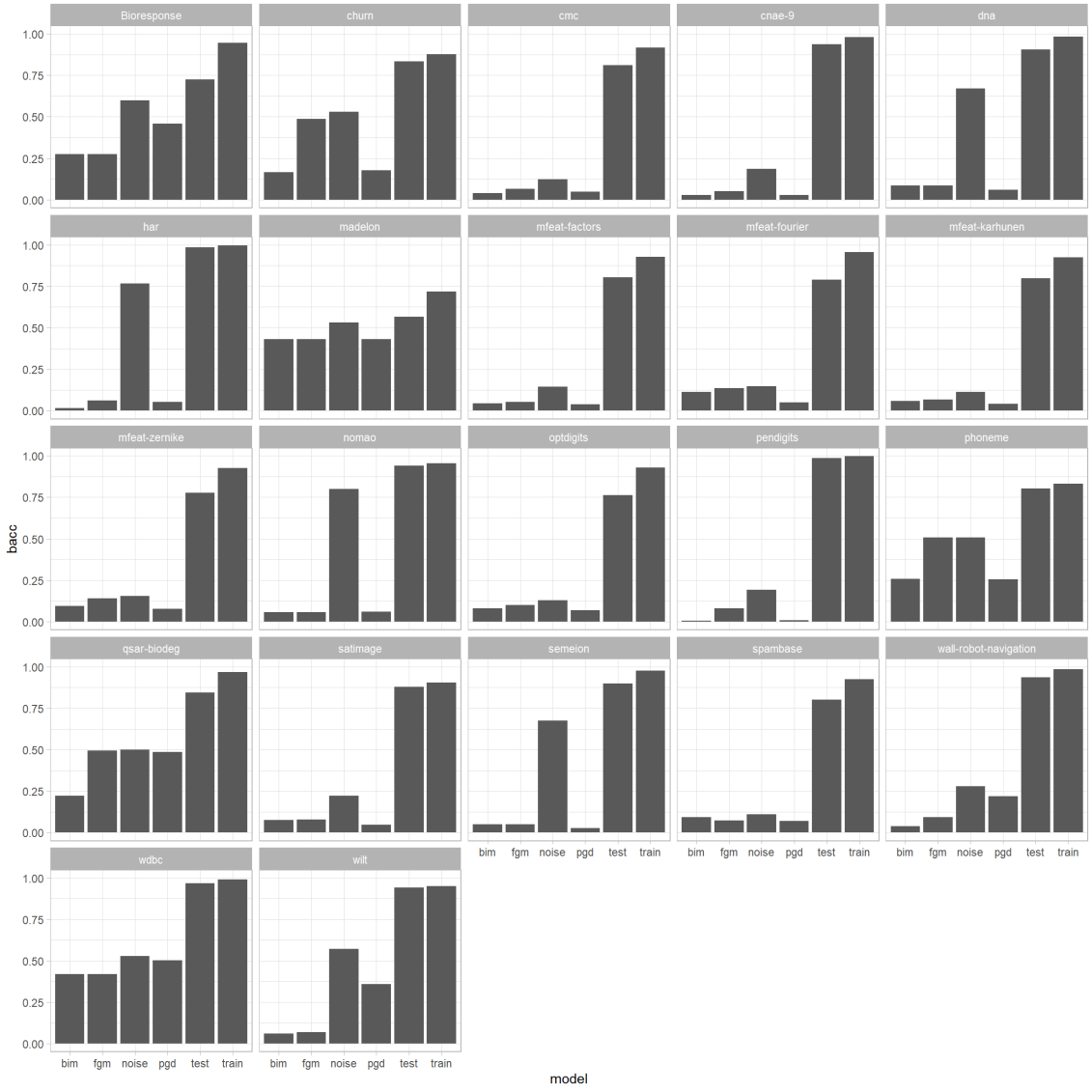
Przygotowane w ten sposób klasyfikator poddano atakom. Wydzieloną (i nieużywaną do trenowania ani walidacji klasyfikatora) część zbiorów danych poddano zaburzeniom zgodnie z procedurą opisaną w wybranych atakach. Do implementacji ataków użyto pakietu ART [31].

Jako dodatkowy strumień danych, traktowany jako czwarty *atak pozorny*, wygenerowano dane zaburzone w sposób losowy - poprzez dodanie do nich szumu gaussowskiego o amplitudzie zgodnej z rzędem amplitudy zaburzeń generowanych przez rzeczywiste ataki. Celem było stwierdzenie odporności badanego klasyfikatora na dane zaburzone w sposób naturalny - na ile potrafi on rozróżniać ataki od *ataków pozornych* w scenariuszu zbliżonym do rzeczywistego. Szum gaussowski służy w tym przypadku jako odniesienie - pokazanie jak model zachowa się przy losowych, nieukierunkowanych zaburzeniach danych wejściowych. W przeciwieństwie do przykładów adwersaryjnych, celowo skonstruowanych w celu wprowadzenia błędów w działaniu modelu, szum gaussowski reprezentuje naturalne, przypadkowe fluktuacje danych. Jeżeli model wykazuje znacząco większą wrażliwość na przykłady adwersaryjne niż na szum gaussowski o porównywalnej wielkości, świadczy to o tym, że atak adwersaryjny efektywnie wykorzystuje specyficzne słabości modelu, a nie jedynie jego ogólną wrażliwość na perturbacje danych wejściowych. W przypadku gdyby nie było istotnych różnic pomiędzy spadkiem jakości działania modelu wywołanym szumem a tym wywołanym przez ataki, może to świadczyć o słabości samych modeli - ich nadmiernej czułości na dowolne modyfikacje danych wejściowych.

Podsumowanie wyników trenowania, oraz zmiany jakości wyników pracy modeli po przeprowadzonych atakach rzeczywistych i ataku pozornym, przedstawiono w Tabeli 4.12, oraz w formie graficznej na rysunku 4.8. Kolumna ”train” została dodana dla celów poglądowych - wyniki klasyfikacji po dokonaniu ataku oraz po nałożeniu szumu należy porównywać z kolumną przedstawiającą jakość klasyfikacji sprawdzaną na zbiorze ”test”.

Tabela 4.12: Balanced accuracy dla modelu sieci neuronowej oraz modelu sieci po przeprowadzeniu ataków i wprowadzeniu szumu losowego.

zbiór danych	BIM	FGM	szum	PGD	test	train
Bioresponse	0.27	0.27	0.60	0.46	0.72	0.95
churn	0.17	0.49	0.53	0.18	0.83	0.88
cmc	0.04	0.07	0.12	0.05	0.81	0.92
cnae-9	0.03	0.05	0.19	0.03	0.94	0.98
dna	0.09	0.09	0.67	0.06	0.91	0.98
har	0.01	0.06	0.77	0.05	0.99	1.00
madelon	0.43	0.43	0.53	0.43	0.57	0.72
mfeat-factors	0.04	0.05	0.14	0.04	0.80	0.93
mfeat-fourier	0.11	0.14	0.15	0.05	0.79	0.96
mfeat-karhunen	0.06	0.07	0.11	0.04	0.80	0.92
mfeat-zernike	0.09	0.14	0.15	0.08	0.78	0.93
nomao	0.06	0.06	0.80	0.06	0.94	0.96
optdigits	0.08	0.10	0.13	0.07	0.76	0.93
pendigits	0.00	0.08	0.19	0.01	0.99	1.00
phoneme	0.26	0.51	0.51	0.26	0.80	0.83
qsar-biodeg	0.22	0.50	0.50	0.49	0.84	0.97
satimage	0.08	0.08	0.22	0.05	0.88	0.91
semeion	0.05	0.05	0.68	0.03	0.90	0.98
spambase	0.09	0.07	0.11	0.07	0.80	0.93
wall-robot-navigation	0.04	0.09	0.28	0.22	0.94	0.99
wdbc	0.42	0.42	0.53	0.50	0.97	0.99
wilt	0.06	0.07	0.57	0.36	0.94	0.95



Rys. 4.8: Graficzna reprezentacja wyników prezentowanych w Tabeli 4.12.

Następnie uruchomiono procedurę badania modeli i wykrywania ataków analogiczną jak we wcześniejszych częściach eksperymentu - stworzenie modeli zastępczych i wyznaczenie wartości atrybutów diagnostycznych.

Otrzymane wartości atrybutów diagnostycznych użyto następnie jako dane wejściowe do wytrenowanych w poprzedniej części eksperymentu dwóch klasyfikatorów wykrywających ataki. Ponieważ przy trenowaniu tych klasyfikatorów nie były użyte ataki FGM, PGD oraz BIM, uzyskaną jakość klasyfikacji można traktować jako miarę zdolności klasyfikatorów do generalizacji i wykrywania nieznanych wcześniej ataków.

Wyniki zbiorcze przedstawia tabela 4.13.

Tabela 4.13: Balanced accuracy i Kappa Cohena dla klasyfikatorów wykrywających dowolny z ataków FGM, PGD oraz BIM.

Klasyfikator	random forest	XGBoost
balanced accuracy	0.95	0.91
Kappa Cohena	0.94	0.80

Oba klasyfikatory potwierdziły swoją skuteczność dla nowych, niewystępujących w danych uczących, typów ataków, oraz dla nowego typu badanego modelu uczenia maszynowego.

Analizując dane szczegółowe - klasyfikator oparty o random forest w dwóch przypadkach błędnie zaklasyfikował zbiór oryginalny jako atak (Tabela 4.14).

Tabela 4.14: Błędne predykcje klasyfikatora opartego o random forest.

Indeks	**Zbiór danych**	**Model**	**Atak**	**Atak - binarnie**	**Przewidywanie**
33	madelon	nn	org	0	1
43	mfeat-fourier	nn	org	0	1

Dla klasyfikatora opartego o XGBoost wystąpiło inne zjawisko - pomylenie danych zaszumionych z danymi oryginalnymi. Tego typu pomyłka wystąpiła w przypadkach zbiorów dla których poziom zaszumienia nie wpłynął znacząco na balanced accuracy klasyfikacji. Przypadki błędnej klasyfikacji przedstawiono w Tabeli 4.15.

Tabela 4.15: Błędne predykcje klasyfikatora opartego o XGBoost.

Indeks	**Zbiór danych**	**Model**	**Atak**	**Atak - binarnie**	**Przewidywanie**
2	Bioresponse	nn	noise	1	0
3	Bioresponse	nn	org	0	1
22	dna	nn	noise	1	0
23	dna	nn	org	0	1
27	har	nn	noise	1	0
33	madelon	nn	org	0	1
57	nomao	nn	noise	1	0

4.2.6 Aktualizacja klasyfikatorów

Kolejnym krokiem było włączenie nowych ataków do zbioru uczącego i aktualizacja klasyfikatorów.

Nowy zbiór uczący zawierał dane z następującej liczby kombinacji klasyfikatorów i modeli - Tabela 4.16.

Tabela 4.16: liczba zbiorów danych dla kombinacji atak-model atakowany.

Atak	LR	sieć neuronowa	SVM	XGBoost
Brak ataku (dane oryginalne)	22	22	22	22
BIM		22		
FGM		22		
HopSkipJump	22		22	20
Szum Gaussowski		22		
PermuteAttack	22		22	22
PGD		22		
ZOO	22		22	22

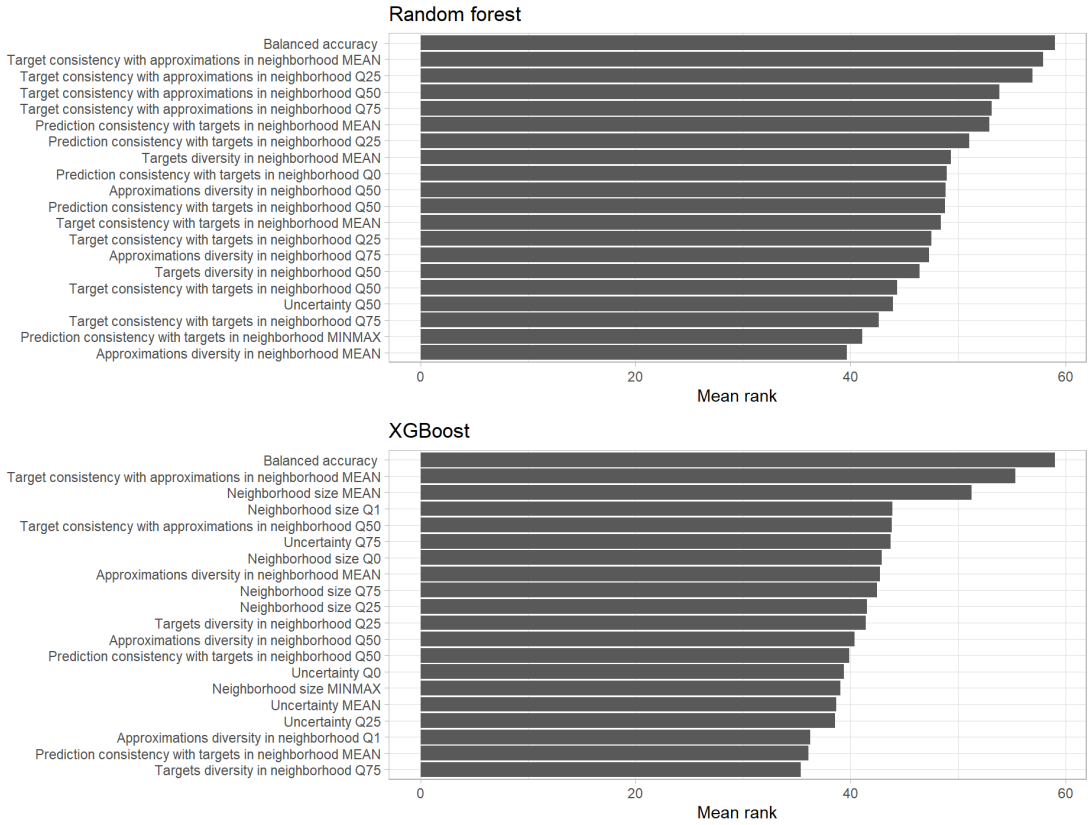
Ponownie wykonano trenowanie klasyfikatorów, wykorzystując scenariusze używane w poprzednich sekcjach. Wyniki przedstawiono w Tabeli 4.17.

Tabela 4.17: Balanced accuracy oraz odchylenie standardowe dla każdego ze scenariuszy treningu klasyfikatorów wykrywających fakt ataku - na zaktualizowanych danych.

Scenariusz	random forest	XGBoost
10-krotna walidacja krzyżowa	0.9212 (0.07)	0.931 (0.0676)
Usunięcie jednego ataku	0.9784 (0.037)	0.9654 (0.0515)
Usunięcie jednego zbioru danych	0.9156 (0.1497)	0.9594 (0.1102)
Usunięcie jednego modelu	0.9409 (0.0215)	0.96 (0.0346)

Powtórzono analizę istotności atrybutów diagnostycznych. Wyniki potwierdziły największy wpływ balanced accuracy, oraz podobną jak we wcześniejszym procesie trenowania grupę najistotniejszych atrybutów dla poszczególnych klasyfikatorów. 4.9

Analogiczne kroki przeprowadzono dla klasyfikatorów służących zadaniu izolacji ataków. Wyniki przedstawiono w Tabeli 4.18 oraz na Rysunku 4.10.

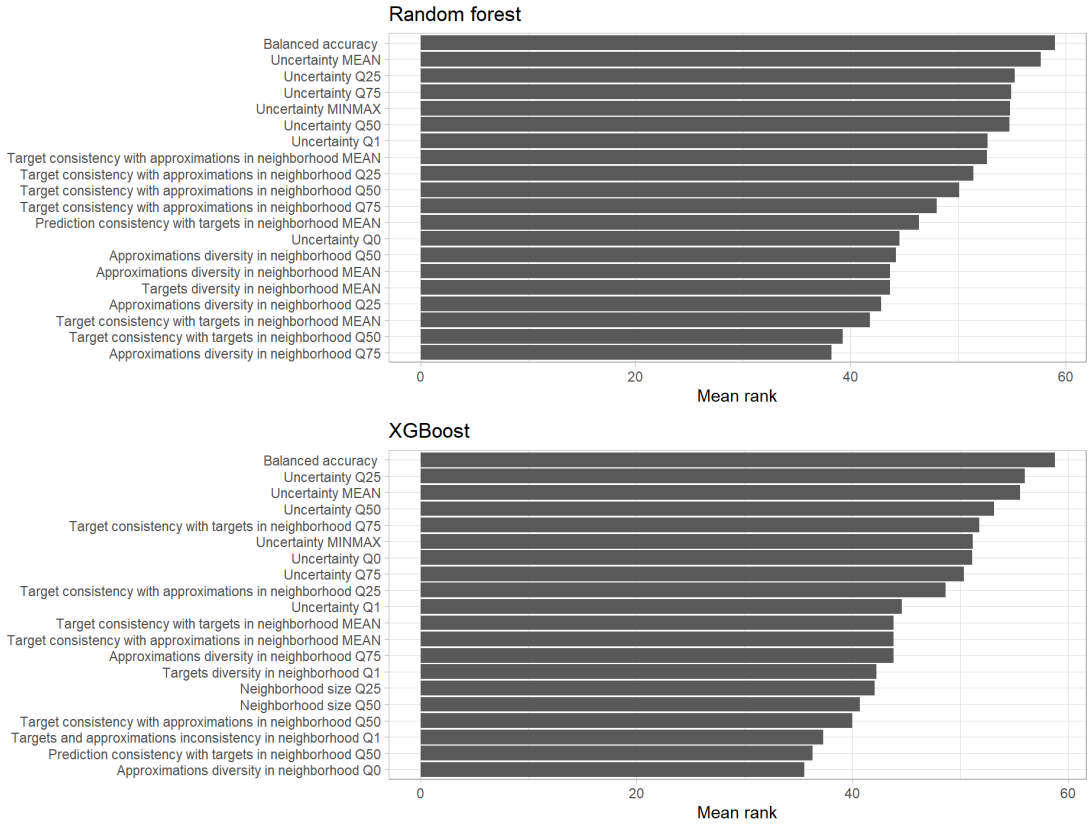


Rys. 4.9: Średnia ranga istotności atrybutów diagnostycznych używanych przez klasyfikatory wykrywające fakt ataku.

Tabela 4.18: Balanced accuracy oraz odchylenie standardowe dla każdego ze scenariuszy treningu klasyfikatorów wykrywających typ ataku - na zaktualizowanych danych.

Scenariusz	random forest	XGBoost
10-krotna walidacja krzyżowa	0.6125 (0.1338)	0.6678 (0.1652)
Usunięcie jednego zbioru danych	0.7277 (0.1841)	0.724 (0.2035)
Usunięcie jednego modelu	0.5938 (0.2902)	0.5778 (0.2879)

Również i w tym przypadku nie udało się uzyskać skutecznego klasyfikatora wykrywającego typ ataków.



Rys. 4.10: Średnia ranga istotności atrybutów diagnostycznych używanych przez klasyfikatory wykrywające typ ataku.

Aktualizacja klasyfikatorów bez balanced accuracy

W poprzednich sekcjach, dla każdego z trenowanych klasyfikatorów wykrywających ataki i ich typ, najbardziej istotnym atrybutem diagnostycznym wpływającym na działanie klasyfikatora była wartości balanced accuracy uzyskiwana przez model poddawany diagnostyce. Jest to problematyczna co najmniej z trzech powodów.

- W rzeczywistych zastosowaniach proporcja przykładów adwersaryjnych do normalnych danych jest zazwyczaj bardzo niska. Balanced accuracy, poprzez równe ważenie czułości i swoistości, może dawać zbyt optymistyczny obraz skuteczności klasyfikatora w scenariuszach w których atakujący selektywnie wpływa na wyniki, bez znaczącego spadku parametru balanced accuracy.

- Balanced accuracy traktuje wszystkie błędy klasyfikacji jednakowo, podczas gdy w kontekście bezpieczeństwa koszty błędów nie muszą być symetryczne. Niewykryty atak jest zwykle bardziej kosztowny niż fałszywy alarm.
- Balanced accuracy nie uwzględnia stopnia pewności klasyfikatora co do swoich predykcji. W kontekście bezpieczeństwa często potrzebujemy więcej informacji niż tylko binarną decyzję.

Biorąc powyższe pod uwagę, wytrenowano klasyfikatory wykrywające ataki bez włączania balanced accuracy jako atrybutu diagnostycznego. Taka procedura pozwoli również ocenić w jakim stopniu utrata informacji o wartości balanced accuracy wpłynie na zdolność wykrywania faktu i rozróżniania typu ataku.

Wyniki trenowania klasyfikatorów dla zagadnień wykrywania faktu oraz wykrywania typu ataku przedstawiono w Tabelach 4.19 oraz 4.20.

Tabela 4.19: Balanced accuracy oraz odchylenie standardowe dla każdego ze scenariuszy treningu klasyfikatorów wykrywających fakt ataku - bez balanced accuracy.

Scenariusz	random forest	XGBoost
10-krotna walidacja krzyżowa	0.8741 (0.1082)	0.9085 (0.0787)
Usunięcie jednego ataku	0.9653 (0.0544)	0.9673 (0.0442)
Usunięcie jednego zbioru danych	0.8632 (0.1815)	0.9142 (0.135)
Usunięcie jednego modelu	0.9284 (0.0201)	0.9077 (0.0981)

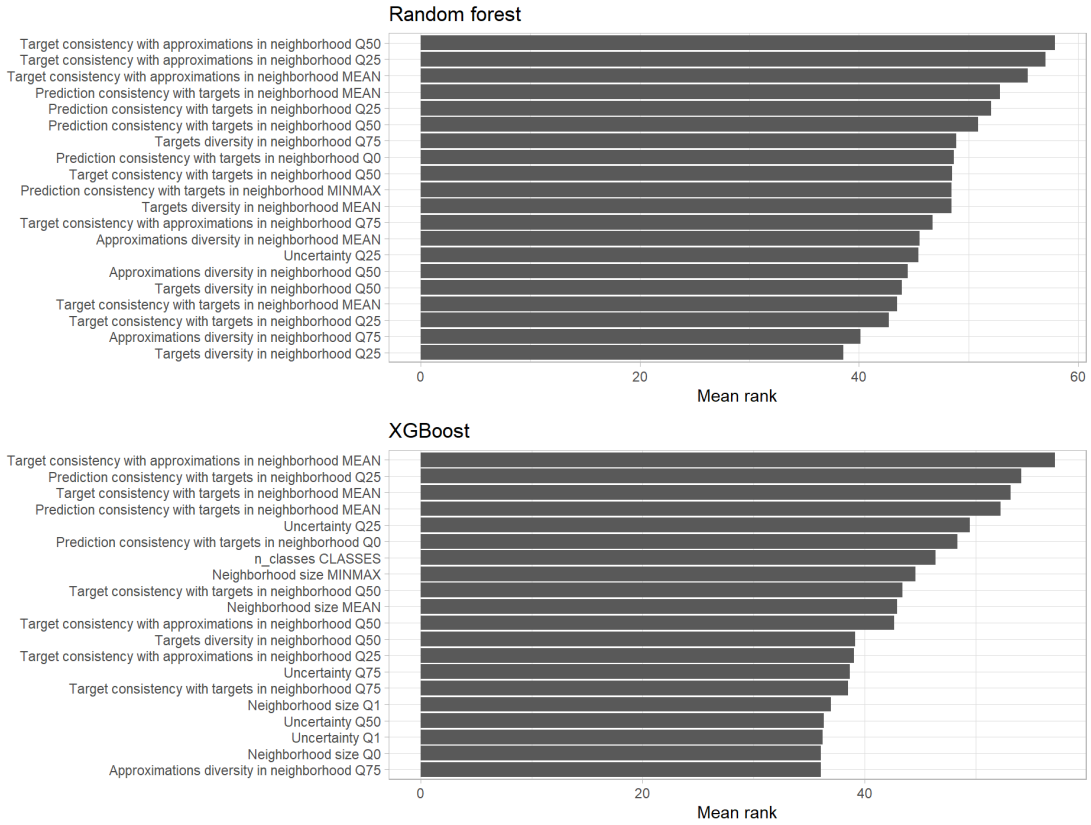
Tabela 4.20: Balanced accuracy oraz odchylenie standardowe dla każdego ze scenariuszy treningu klasyfikatorów wykrywających typ ataku - bez brania pod uwagę cechy balanced accuracy.

Scenariusz	random forest	XGBoost
10-krotna walidacja krzyżowa	0.5885 (0.1433)	0.5933 (0.172)
Usunięcie jednego zbioru danych	0.6368 (0.251)	0.6482 (0.2472)
Usunięcie jednego modelu	0.5236 (0.2412)	0.5102 (0.24)

Otrzymane na tym etapie klasyfikatory wykrywające ataki zostały uznane za wystarczająco dobre do ich praktycznego zastosowania. Przy najbardziej korzystnym scenariuszu trenowania klasyfikatora, z usunięciem jednego ataku, usunięcie parametru balanced accuracy nie wpłynęło w sposób istotny na jakoś predykcji klasyfikatora

XGBoost (0.9654 (odch. std. 0.0515) z BA vs 0.9673 (odch. std. 0.0442) bez BA). Jakość klasyfikatora opartego o RF zmieniła się z 0.9784 (odch. std. 0.037) na 0.9653 (odch. std. 0.0544).

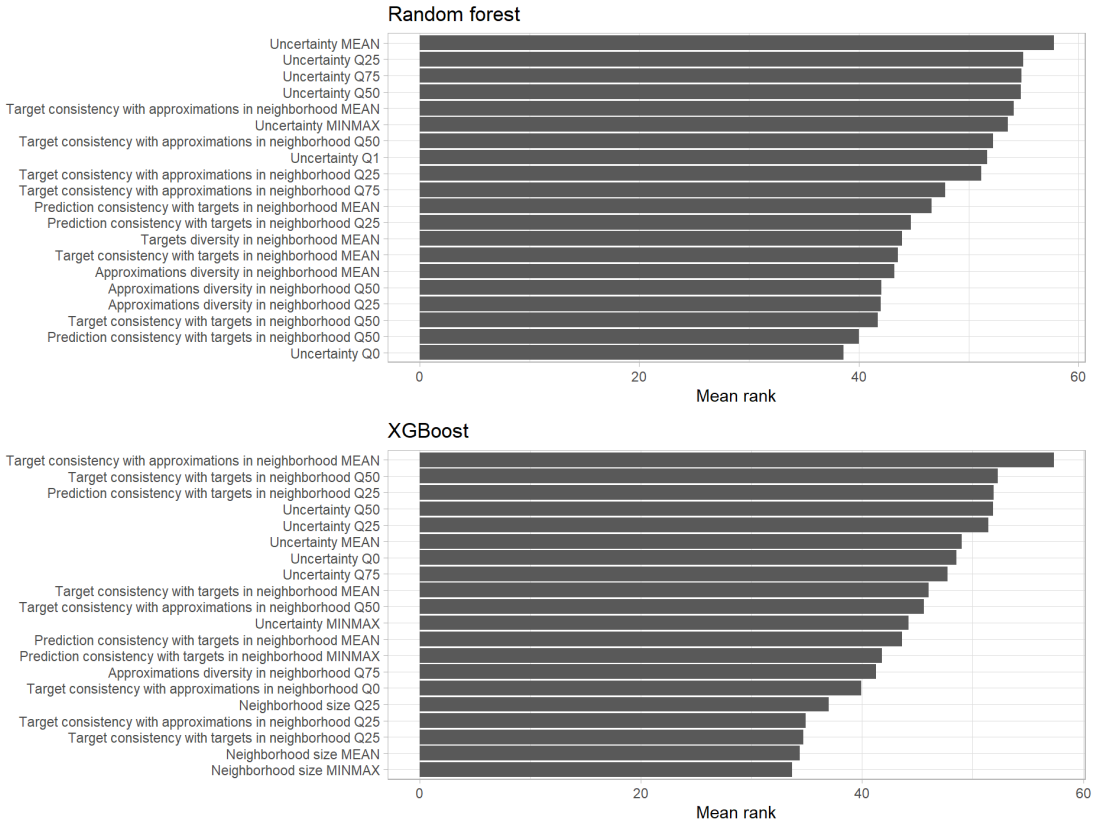
Sprawdzono również jak usunięcie parametru balanced accuracy wpłynęło na istotność pozostałych atrybutów diagnostycznych. Wyniki przedstawiono na Rysunkach 4.11 i 4.12.



Rys. 4.11: Średnia ranga istotności atrybutów diagnostycznych dla obu danych klasyfikatorów wykrywających fakt ataku - bez balanced accuracy.

Dla klasyfikatorów opartych o random forest nie wystąpiła znacząca zmiana istotności cech branych pod uwagę przy klasyfikacji. Natomiast w przypadku XGBoost ranking ten uległ znaczącej zmianie. Jest to zgodne z przewidywaniami, opartymi o zrozumienie sposobu działania tych modeli. W przypadku random forest:

- Każde drzewo w lesie jest budowane niezależnie od innych.
- Cechy są wybierane losowo na każdym poziomie podziału.



Rys. 4.12: Średnia ranga istotności atrybutów diagnostycznych dla obu danych klasyfikatorów wykrywających typ ataku - bez balanced accuracy.

- Istotność cechy jest zazwyczaj obliczana poprzez uśrednienie jej wpływu we wszystkich drzewach.
- Usunięcie jednej cechy nie wpływa znacząco na względną istotność pozostałych, ponieważ każda z nich była oceniana niezależnie.

Innymi słowy - gdy istnieją skorelowane cechy, usunięcie jednej z nich nie ma dużego wpływu, ponieważ inna skorelowana cecha może przejąć jej rolę. Jeśli najważniejsza cecha ma silnie skorelowane cechy ”zastępcze”, jej usunięcie nie zmienia rankingów, ponieważ model automatycznie wybiera je do podziału.

Z kolei dla modelu XGBoost:

- Drzewa są budowane sekwencyjnie, gdzie każde kolejne koncentruje się na naprawianiu błędów poprzednich (boosting).
- Istotność cech jest silnie powiązana z kolejnością ich wykorzystania w sekwencji.

- Po usunięciu najważniejszej cechy, model musi znaleźć nowy sposób minimalizacji błędu.
- To prowadzi do znaczącej reorganizacji wykorzystania pozostałych cech, co skutkuje zmianą ich względnej istotności.

W XGBoost, ponieważ podziały są deterministyczne, usunięcie ważnej cechy zmusza model do znalezienia nowej cechy. Gdy cecha ma unikalne informacje (brak silnych korelacji z innymi cechami), usunięcie jej zmusza XGBoost do przebudowy całego procesu podziału, przez co ranking cech może znacząco się zmienić.

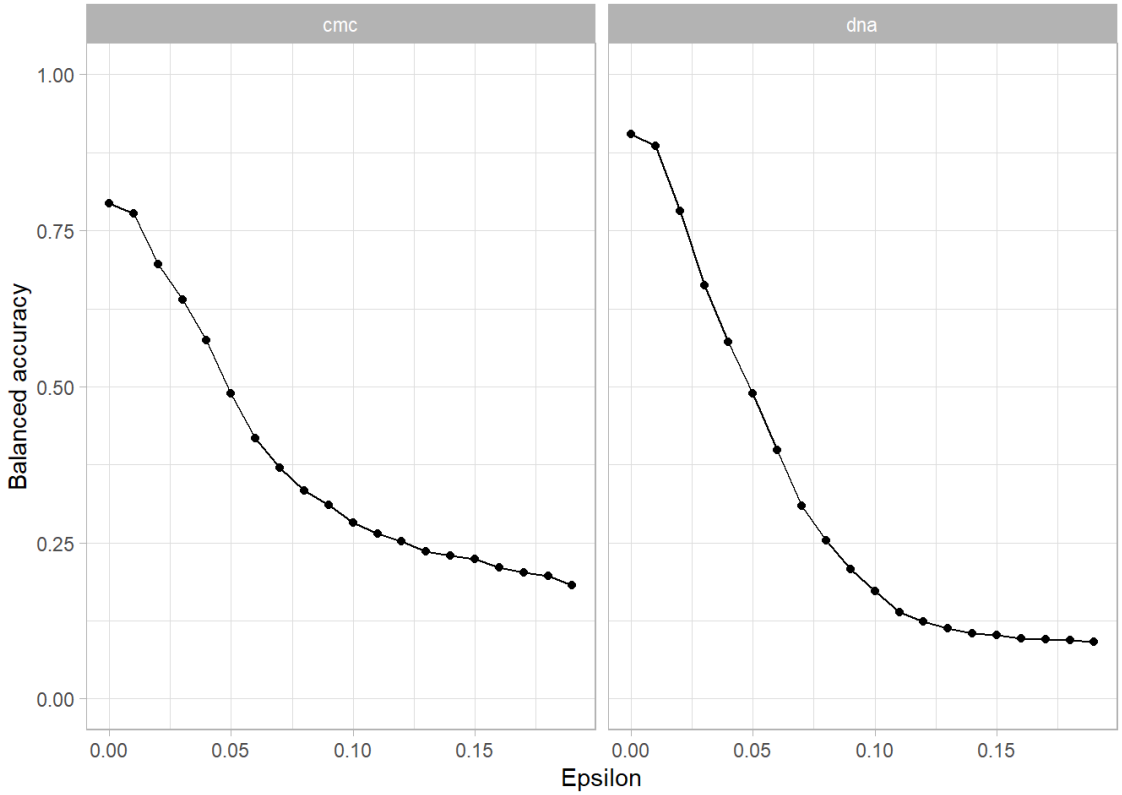
4.2.7 Weryfikacja skuteczności klasyfikatora w funkcji parametru intensywności ataku

W testowanych dotychczas przypadkach ataku zakładano standardowy, zadany przez autora danej techniki ataku, współczynnik skali perturbacji ϵ. Praktyczne znaczenie tego współczynnika jest istotne z punktu widzenia zarówno strony atakującej jak i próbującej wykryć atak. Jego obniżenie będzie powodować mniejsze, trudniejsze do wykrycia zaburzenia, przy jednoczesnym mniejszym wpływie na poprawność działania systemu atakowanego. Jest to szczególny przypadek problemu ogólnego, w którym każdy atak zakłada jakiś poziom świadomej równowagi pomiędzy jego skutecznością a trudnością w wykryciu [34].

W celu sprawdzenia zależności skuteczności ataku od jego skali, dla dwóch zbiorów danych, cmc i dna, przeprowadzono punktową analizę sprawdzającą jak zmienia się miara balanced accuracy w zależności od wartości ϵ w ataku FGM. Weryfikacji poddano 20 wartości ϵ z zakresu od 0 do 0.2 z krokiem równym 0.01. Wyniki, pokazujące jak skutecznie atak jest w stanie obniżyć skuteczność modelu atakowanego, przedstawiono na rysunku 4.13.

Po tym sprawdzeniu, przygotowano analogiczny zestaw ataków typu FGM dla każdego ze zbiorów - w sumie 20 ataków z różną intensywnością na każdy z 22 zbiorów - po czym sprawdzono skuteczność wykrywania ataków przez poprzednio przygotowane klasyfikatory. Wyniki zbiorcze przedstawiono w Tabeli 4.21

Następnie powtórzono eksperyment, tym razem do wykrywania ataku wykorzystując klasyfikatory random forest i XGBoost wytrenowane bez używania balanced accuracy jako atrybutu wejściowego. Wyniki przedstawia Tabela 4.22). Wysoka jakość uzyskanej klasyfikacji, wyższa niż przy użyciu klasyfikatorów uwzględniających



Rys. 4.13: Skuteczność ataku (rozumiana jako wartość balanced accuracy zaatakowanego modelu ML) w funkcji parametru skalującego.

Tabela 4.21: Skuteczność klasyfikatorów wykrywających fakt ataku badana na przykładach ataku o różnej skali intensywności.

Metryka pomiaru	random forest	XGBoost
balanced accuracy	0.95	0.98
Kappa Cohena	0.86	0.67

balanced accuracy, może to świadczyć o tym, że przy niskiej intensywności ataku, gdy jeszcze jego ogólna jakość działania nie została w sposób widoczny uwidoczniona przez spadek wartości balanced accuracy atakowanego modelu, pozostałe atrybuty diagnostyczne już pozwalają na skuteczne wykrycie ataku.

Tabela 4.22: Skuteczność klasyfikatorów wykrywających fakt ataku badana na przykładach ataku o różnej skali intensywności - bez uwzględnienia balanced accuracy jako atrybutu diagnostycznego.

Metryka pomiaru	random forest	XGBoost
balanced accuracy	0.95	0.90
Kappa Cohena	1.0	0.91

4.3 Analiza i dyskusja wyników

Przeprowadzone eksperymenty dostarczyły wyników, które warto przeanalizować w kilku odrębnie ujętych aspektach. Poniższa analiza skupia się na tych aspektach, oraz ich implikacjach zarówno dla teorii jak i praktyki związanej z implementacją wyników badań w środowiskach produkcyjnych.

4.3.1 Skuteczność detekcji ataków

Ogólna skuteczność klasyfikatorów

Najważniejszym osiągnięciem przeprowadzonych badań jest potwierdzenie możliwości skutecznego wykrywania ataków adwersaryjnych przy użyciu atrybutów diagnostycznych.

Klasyfikatory osiągnęły wyniki pozwalające na ich bezpośednie zastosowanie w systemach produkcyjnych:

- random forest: balanced accuracy na poziomie 0.87-0.97 (w zależności od scenariusza),
- XGBoost: balanced accuracy na poziomie 0.91-0.97 (w zależności od scenariusza).

Te wysokie wartości utrzymały się nawet po usunięciu parametru balanced accuracy z zestawu cech. Świadczy o stabilności i niezawodności metody. Jest to szczególnie istotne w kontekście praktycznych zastosowań, gdzie nie zawsze możemy mieć dostęp do rzeczywistych etykiet dla oceny jakości klasyfikacji.

Możliwości wykrywania nowych nieznanych typów ataku

Szczególnie wartościowym aspektem praktycznym jest weryfikacja skuteczności klasyfikatorów w wykrywaniu wcześniej nieznanych typów ataków. Jest to jedno z wąskich gardeł większości podejść do wsparcia wykrywania zagrożeń cyberbezpieczeństwa w oparciu o uczenie maszynowe nadzorowane. W proponowanym ujęciu analizie podlegają nie cechy specyficzne poszczególnych ataków, ale cechy danych przez te ataki zmienianych. W przeprowadzonym eksperymencie klasyfikatory wytrenowane na atakach typu *czarna skrzynka* (ZOO, HopSkipJump, PermuteAttack) skutecznie - balanced accuracy od 0.91 (dla random forest) do 0.95 (XGBoost) wykrywały również ataki typu *biała skrzynka* (BIM, PGD, FGM).

Ta zdolność do generalizacji jest bardzo istotna w zastosowaniach praktycznych, gdzie system musi radzić sobie z nowymi, wcześniej nieznanymi typami ataków. W praktyce, dołączenie tych ataków do zbioru uczącego oraz przetrenowanie klasyfikatorów wykrywających ataki następuje z istotnym przesunięciem czasowym względem ich pojawienia się. Innymi słowy - opracowane klasyfikatory są do pewnego stopnia odporne przynajmniej na część możliwych ataków adwersaryjnych "nowych" (ang. zero-day attacks).

4.3.2 Analiza błędów klasyfikacji

Analiza przypadków błędnej klasyfikacji ujawniła interesujące wzorce:

- Klasyfikator wykrywający ataki bazujący na metodzie random forest wykazywał tendencję do fałszywych alarmów w przypadku zbiorów danych o wysokiej złożoności (madelon, mfeat-fourier).
- Klasyfikator wykrywający ataki bazujący na algorytmie XGBoost miał trudności z rozróżnieniem między danymi zaszumionymi a oryginalnymi, szczególnie w przypadkach gdy szum nie wpływał znacząco na jakość klasyfikacji.

Te obserwacje sugerują komplementarność obu podejść i potencjalną wartość w ich łączeniu w systemach produkcyjnych.

4.3.3 Rola atrybutów diagnostycznych

Kluczowe atrybuty

Badania wykazały, że najważniejsze dla wykrywania ataku zagregowane atrybuty diagnostyczne, poza atrybutem balanced accuracy, to:

- Średnia niepewności predykcji (ang. mean uncertainty).
- Spójność celu z przybliżeniami w sąsiedztwie (ang. target consistency with approximations in neighborhood) - średnia oraz pierwsze dwa kwantyle.
- Średnia różnorodność celów i przybliżeń w sąsiedztwie (ang. mean targets diversity in neighborhood oraz mean approximations diversity in neighborhood).
- Średni rozmiar sąsiedztwa (ang. mean neighborhood size).

Co istotne, ranking ten pozostał względnie stabilny nawet po usunięciu parametru balanced accuracy, szczególnie dla klasyfikatora random forest. Dla klasyfikatora opartego o XGBoost, mimo przebudowy drzewa decyzyjnego, jako najważniejsze cechy wciąż występowały cechy pochodne od wyżej wspomnianych atrybutów diagnostycznych.

Sugeruje to możliwy kierunek optymalizacji klasyfikatorów w sytuacji gdy niezbędne będzie polepszenie wydajności kosztem dokładności klasyfikacji - np. w scenariuszach przetwarzania dużych strumieni danych blisko źródła ich powstawania (ang. edge processing).

Różnice między klasyfikatorami wykrywającymi ataki

Interesująca jest obserwacja różnic w sposobie wykorzystania atrybutów przez różne klasyfikatory:

- random forest wykazywał większą stabilność w rankingu istotności cech,
- XGBoost znacząco zmieniał ranking istotności po usunięciu balanced accuracy

Ta różnica wynika zapewne z odmiennych mechanizmów działania obu algorytmów i sugeruje, że każdy z nich może być lepiej dostosowany do różnych scenariuszy użycia.

Praktyczny wniosek wynikający z tej obserwacji jest taki, że w przypadku gdy nie ma ograniczeń wydajnościowych warto w zastosowaniu praktycznym wykorzystać model stanowiący złożenie obu klasyfikatorów - XGBoost i random forest.

4.3.4 Ograniczenia w wykrywaniu typu ataku

Próba stworzenia klasyfikatorów rozróżniających konkretne typy ataków nie przyniosła zadowalających rezultatów:

- balanced accuracy na poziomie 0.59-0.65 dla różnych scenariuszy,
- brak znaczącej poprawy po dodaniu nowych typów ataków do zbioru treningowego

Te wyniki sugerują, że atrybuty diagnostyczne, choć skuteczne w wykrywaniu faktu ataku, nie zawierają wystarczającej informacji do precyzyjnego rozróżnienia jego typu.

Może to wynikać z:

- podobnego wpływu różnych ataków na lokalną strukturę przestrzeni cech,
- zbyt wysokiego poziomu abstrakcji atrybutów diagnostycznych,
- możliwego przekrywania się charakterystyk różnych typów ataków.

4.3.5 Uniwersalność podejścia diagnostycznego

Skuteczność modeli zastępczych

Kluczowym wynikiem badań jest potwierdzenie skuteczności metody diagnostycznej opartej na modelach zastępczych dla szerokiego spektrum modeli ML, reprezentujących:

- modele liniowe (regresja logistyczna),
- modele jądrowe (SVM),
- modele zespołowe (XGBoost),
- sieci neuronowe

Wysokie wartości współczynnika kappa Cohena (>0.9) dla modeli zastępczych względem wszystkich typów modeli diagnozowanych potwierdzają uniwersalność metody. Wskazuje to na możliwość stosowania jednolitego podejścia diagnostycznego niezależnie od architektury diagnozowanego modelu. Dodatkowym aspektem

zweryfikowanym w eksperymentach było zastosowanie metody opartej o tworzenie i analizę atrybutów diagnostycznych. Metoda, opisana w artykule [41], pierwotnie miała służyć wykrywaniu przypadków w których konieczne jest wyjaśnienie błędów pojawiających się w pracy diagnozowanego modelu uczenia maszynowego biorących się z konkretnych właściwości danych uczących (np. niedoreprezentacja cech lub przykładów treningowych) lub procesu uczenia (np. przeuczenie lub niedouczenie). Opracowana w ramach niniejszej pracy metoda diagnostyczna oraz przeprowadzone eksperymenty rozszerzają zakres stosowalności atrybutów diagnostycznych.

Rozróżnienie między atakami a szumem

Jako dodatkowy aspekt badań włączono sprawdzenie zdolności metody do rozróżnienia między celowymi atakami adwersaryjnymi a zjawiskami naturalnymi, niebędącymi wynikiem celowych ataków. Analizując wyniki eksperymentów z szumem gaussowskim, który może być traktowany jako przybliżenie naturalnych zmian w danych, w tym mogących wynikać ze zjawiska concept drift, zaobserwowano:

Ataki adwersaryjne generują charakterystyczne wzorce w atrybutach diagnostycznych, szczególnie w:

- niepewności predykcji,
- spójności lokalnej struktury danych,
- rozkładzie odległości w przestrzeni cech.

Naturalne zmiany (symulowane szumem gaussowskim) wykazują:

- bardziej równomierny wpływ na wszystkie atrybuty diagnostyczne,
- mniejszy wpływ na spójność lokalną,
- zachowanie podstawowych właściwości statystycznych danych.

Ta różnica w charakterystykach pozwala na wnioskowanie o możliwości zastosowania proponowanej metody do wykrywania zjawiska tzw. *concept drift* w danych, w których zjawisko to może wystąpić. Zanim ten aspekt zostanie w praktyce zastosowany konieczne będzie przeprowadzenie dalszych badań - już poza kontekstem niniejszej pracy.

4.3.6 Implikacje praktyczne

Wyniki badań mają istotne implikacje dla praktycznych zastosowań:

Projektowanie systemów obronnych

- Wielowarstwowa ochrona:
  - Prawdopodobnie warto jest łączyć wyniki różnych klasyfikatorów i metod wykrywających ataki (np. random forest i XGBoost) ze względu na ich komplementarne właściwości.
  - Warto rozważyć implementację systemu wczesnego ostrzegania bazującego na niepewności predykcji.
- Adaptacyjność:
  - System powinien być regularnie aktualizowany nowymi przykładami wykrytych i zweryfikowanych lub opisanych w literaturze ataków, choć badania pokazują, że nawet bez tego zachowuje wysoką skuteczność.
  - Warto rozważyć wprowadzenie mechanizmów uczenia aktywnego, lub innych technik częściowo nadzorowanych (ang. semi-supervised learning) w celu regularnej aktualizacji zbioru uczącego i dotrenowywania klasyfikatorów wykrywających.
- Monitorowanie niepewności: Szczególną uwagę należy zwrócić na monitorowanie niepewności predykcji jako kluczowego wskaźnika potencjalnego ataku.

4.3.7 Podsumowanie

Przeprowadzone badania potwierdzają skuteczność proponowanego podejścia do wykrywania ataków adwersaryjnych, jednocześnie wskazując na jego ograniczenia w kontekście izolacji konkretnych typów ataków. Szczególnie wartościowa jest zdolność metody do wykrywania nowych, nieznanych wcześniej typów ataków, dzięki czemu jest ona użyteczna w praktycznych zastosowaniach. Wyniki sugerują, że połączenie różnych typów klasyfikatorów może prowadzić do stworzenia bardziej odpornych systemów obronnych. Badania wykazały również uniwersalność metody diagnostycznej opartej na modelach zastępczych, potwierdzając jej skuteczność dla różnych architektur modeli uczenia maszynowego. Zdolność do rozróżniania między celowymi atakami

a naturalnymi zmianami w danych dodatkowo potwierdza praktyczną użyteczność metody.

5 Podsumowanie

W niniejszej pracy przedstawiono prace badawcze służące opracowaniu i wdrożeniu użytecznego klasyfikatora, zdatnego do działania wykrywania ataków adwersaryjnych na różnorodne modele uczenia maszynowego służące klasyfikacji danych tabelarycznych.

Motywacją do podjęcia badań była obserwowana w praktyce luka w zabezpieczeniach systemów wykorzystujących modele uczenia maszynowego w procesach podejmowania decyzji. Luka ta polega na nieuwzględnianiu,przy projektowaniu zabezpieczeń systemów, podatności modeli uczenia maszynowego na specyficzne ataki adwersaryjne, w szczególności te wykorzystujące celowe, minimalne modyfikacje danych wejściowych. Dodatkową motywacją był brak rozwiniętych metod wykrywania ataków działających w paradygmacie czarnej skrzynki oraz niedoreprezentowanie w literaturze badań nad atakami na modele przetwarzające dane tabelaryczne.

Głównym celem pracy było opracowanie, weryfikacja i implementacja metody wykrywania ataków adwersaryjnych na modele uczenia maszynowego trenowane na danych tabelarycznych. Metoda miała działać w reżimie czarnej skrzynki, bez szczegółowych założeń o ani dostępu do architektury i parametrów diagnozowanego modelu.

W ramach realizacji głównego celu pracy postawiono szereg celów szczegółowych.

- Implementacja metod ataku na modele klasyfikujące dane tabelaryczne - tak by w wiarygodny sposób odzwierciedlić możliwe scenariusze ataku i zapewnić wiarygodną ocenę skuteczności badanej metody wykrywania.

- Opracowanie i wdrożenie metody przybliżającej działanie modelu poddawanego diagnostyce - tak by być w stanie objąć monitoringiem modele w scenariuszu braku dostępu do modelu - oraz weryfikacja możliwości pozyskania w ten sposób istotnych atrybutów diagnostycznych charakteryzujących pracę modelu diagnozowanego.

- Opracowanie i weryfikacja klasyfikatorów wykrywających ataki, bazujących na atrybutach diagnostycznych pozyskanych w ramach monitorowania działania diagnozowanego modelu ML.
- Weryfikacja działania opracowanej metody w wykrywaniu ataków adwersaryjnych innych niż te wykorzystywane podczas tworzenia klasyfikatorów wykrywających ataki - badanie zdolności metody do generalizacji.

Zaproponowana metoda będąca przedmiotem badań w niniejszej pracy dedykowana jest dla modeli klasyfikacyjnych przetwarzających dane tabelaryczne. Umożliwia ona wykrywanie ataków adwersaryjnych mających wywołać błędną klasyfikację danych - na etapie inferencji (ataki typu *evasion*). Wymaga ona:

- dostępu do danych wejściowych oraz odpowiadających im decyzji podejmowanych przez diagnozowany model (model ML może być traktowany jako czarna skrzynka),
- możliwości uruchomienia równoległego do diagnozowanego modelu strumienia przetwarzania danych - służącego do obliczania atrybutów diagnostycznych i wykrywania ataków.

Metoda bazuje na dwóch kluczowych elementach: modelu zastępczym aproksymującym działanie diagnozowanego modelu oraz ekstrahowanym z modelu zastępczego zestawie atrybutów diagnostycznych odzwierciedlających zachowanie modelu w kontekście lokalnych otoczeń analizowanych przykładów. Na podstawie tych atrybutów, wytrenowany klasyfikator wykrywa przypadki ataku dla zadanego okna danych.

W ramach prac eksperymentalnych:

- Przygotowano kompleksowy zestaw eksperymentów obejmujący 22 zbiory danych, 3 typy modeli uczenia maszynowego (regresja logistyczna, SVM, XGBoost) oraz 3 różne metody ataku (ZOO, HopSkipJump, PermuteAttack).
- Wykorzystano i zaadaptowano metodykę BrightBox do tworzenia modeli zastępczych, osiągając wysoką zgodność z modelami diagnozowanymi (współczynnik kappa Cohena > 0.9).
- Potwierdzono statystyczną korelację pomiędzy wartościami atrybutów diagnostycznych a występowaniem ataku.

- Stworzono skuteczne klasyfikatory wykrywające ataki (balanced accuracy 0.87-0.97), działające wyłącznie na podstawie wartości zagregowanych atrybutów diagnostycznych generowanych przez modele zastępcze.
- Zweryfikowano zdolność klasyfikatorów do wykrywania nowych (niewystępujących w danych uczących) typów ataków - balanced accuracy 0.91-0.95 dla nowych trzech ataków (BIM, PGD, FGM), atakujących nowy typ modelu uczenia maszynowego (sieci neuronowe), potwierdzając zdolność metody do generalizacji.
- Sprawdzono możliwość wykorzystania metody do rozróżniania typów ataków - osiągnięte wynik negatywny, przy balanced accuracy poniżej 0.65. Wskazuje to na słabą zdolność izolacji typów ataków w przyjętej metodzie.
- Zweryfikowano pozytywnie odporność metody na brak informacji o ocenie jakości działania modelu diagnozowanego - klasyfikatory wykrywające ataki wykrywały je również bez uwzględniania cechy balanced accuracy.
- Zbadano punktowo, ze skutkiem pozytywnym, zdolność metody do odróżniania ataków od szumu.
- Zbadano skuteczność metody w funkcji intensywności ataku - metoda wykazała wysoką skuteczność również dla ataków o niewielkim stopniu zaburzenia.

Zaproponowana metoda:

- Umożliwia wykrywanie celowych zmian danych, służących zaburzaniu działania modeli uczenia maszynowego przetwarzających dane tabelaryczne.
- Działa w reżimie czarnej skrzynki, wymagając jedynie dostępu równoległego z modelem diagnozowanym do jego danych wejściowych oraz wyników predykcji.
- Metoda nie wymaga natomiast:
  - znajomości architektury diagnozowanego modelu,
  - dostępu do parametrów i gradientów modelu diagnozowanego,
  - dostępu do prawdopodobieństw klas (wystarczające są same decyzje - etykiety klas),
  - informacji o ewentualnych wcześniejszych atakach na diagnozowany model,

  - wiedzy o potencjalnych słabościach modelu diagnozowanego.

Ograniczenia metody:

- Metoda, w swoim obecnym kształcie, nie nadaje się do rozróżniania typów ataków.
- Metoda została przygotowania i zweryfikowana jedynie na modelach służących klasyfikacji danych tabelarycznych.
- Metoda służy wykrywaniu ataków na modele na etapie inferencji - nie służy wykrywaniu ataków typu *poisoning*, mających na celu modyfikację granic decyzyjnych modelu w późniejszym czasie.
- Na etapie przygotowywania modelu zastępczego i wyznaczania wartości referencyjnych dla atrybutów diagnostycznych, działanie modelu diagnozowanego traktowane jest jako niezaburzone. Może to być osiągnięte przez użycie danych na których model był trenowany i walidowany, lub przez zapewnienie pozasystemowe (np. wiedzę ekspercką).
- Prace nad metodą prowadzone były przy założeniu niewystępowania zjawiska *concept drift*. Ma to wymiar praktyczny - w rzeczywistych wdrożeniach naturalne jest występowanie zmian w rozkładzie danych w czasie. Decyzja o niewłączeniu badania *concept drift* do zakresu badań była świadoma. W aktualnej wersji metody przyjęto założenie, że model diagnozowany jest wiarygodny w kontekście aktualnych danych, oraz że odpowiedzialność za monitoring i obsługę zjawiska *concept drift* leży po stronie właściciela modelu, który w przypadku wykrycia istotnych zmian w rozkładzie danych, aktualizuje model i przekazuje zaktualizowane dane treningowe do systemu diagnostycznego.

  Wykrywanie *concept driftu* może stanowić jeden z przyszłych kierunków rozwoju metody. Możliwe kierunki rozwoju metody w tym kontekście to:

  - Rozszerzenie metody o mechanizmy wykrywania i rozróżniania pomiędzy atakami adwersaryjnymi a naturalnymi zmianami w danych (*concept drift*)
  - Wprowadzenie procedur automatycznej adaptacji modeli zastępczych i klasyfikatorów wykrywających ataki w odpowiedzi na *concept drift*
  - Opracowanie protokołu aktualizacji wszystkich komponentów metody (model zastępczy, atrybuty diagnostyczne, klasyfikatory) w sytuacji gdy

właściciel modelu diagnozowanego dostarcza jego zaktualizowaną wersję wraz z nowymi danymi treningowymi

– Badanie wpływu zjawiska *concept drift* na skuteczność wykrywania ataków adwersaryjnych.

Wszystkie postawione cele szczegółowe zostały osiągnięte w toku prowadzonych badań, przy jednoczesnym spełnieniu założeń głównego celu pracy. Potwierdzono skuteczność i praktyczną użyteczność zaproponowanej metody w kontekście wykrywania ataków adwersaryjnych na modele uczenia maszynowego przetwarzające dane tabelaryczne. Zakres prac został rozszerzony o badania mające zweryfikować dodatkowe aspekty użytkowe metody, i mogące stanowić podstawę do dalszych prac badawczo-rozwojowych.

5.1 Prace w kontekście działań partnera biznesowego

Prace stanowiły rozwinięcie prowadzonych przez partnera biznesowego, QED Software, prac badawczo-rozwojowych służących otrzymaniu użytecznej metody diagnostyki modeli uczenia maszynowego z wykorzystaniem modeli zastępczych oraz analizy uzyskiwanych za ich pomocą atrybutów diagnostycznych. Ten strumień prac zaowocował stworzeniem biblioteki BrightBox [41], która znalazła zastosowanie przy wdrażaniu monitoringu oraz audytu modeli uczenia maszynowego, i stanowi element stanowiący o przewadze konkurencyjnej partnera biznesowego. Przy pracach nad systemem BrightBox pojawiły się różne hipotezy związane z możliwością rozszerzenia wykorzystania metody o nowe przypadki użycia, nieprzewidziane w czasie projektowania metody analizy modeli zastępczych i atrybutów diagnostycznych. Jednym z takich hipotetycznych zastosowań było wykorzystanie tej metody do wykrywania ataków adwersaryjnych. Istotność tego zagadnienia wzrosła wraz z pojawiającymi się w rozmowach z klientami i partnerami biznesowymi pytaniami o możliwość detekcji ataków dostosowanych do coraz szerzej wykorzystywanych mechanizmów automatycznego podejmowania decyzji bazujących na modelach uczenia maszynowego. Stanowiło to podstawę do podjęcia prac badawczych przedstawionych w niniejszej pracy. Wyniki prac zostały pozytywnie oceniona i odebrana przez partnera biznesowego.

5.2 Rekomendacje implementacyjne

Przy wdrażaniu wyników prac w środowiskach produkcyjnych, należy zwrócić uwagę na kilka aspektów praktycznych - leżących poza zakresem niniejszej pracy, ale z niej wynikających.

Wybór klasyfikatora:

- Random Forest dla systemów wymagających stabilności i interpretiowalności.
- XGBoost dla maksymalizacji skuteczności detekcji.
- Wprowadzenie możliwości przełączania między klasyfikatorami w zależności od kontekstu.
- Połączenie obu podejść w przypadku gdy jest to możliwe bez przekroczenia ograniczeń budżetowych.
- Połączenie ze wstępną selekcją przypadków opartą o ocenę niepewności predykcji.

Optymalizacja wydajnościowa. W przypadku konieczności dostosowania do niskich dostępnych mocy obliczeniowych, lub wysokich wymaganych parametrów szybkości obliczeń, można dokonać optymalizacji konfiguracji atrybutów diagnostycznych:

- Koncentracja na kluczowych atrybutach diagnostycznych (niepewność, spójność, różnorodność) może pozwolić na uproszczenie systemu bez znaczącej utraty skuteczności.
- Wprowadzenie mechanizmu automatycznej selekcji cech, dopasowujących system do potrzeb i specyfiki konkretnej implementacji.

W praktycznych zastosowaniach istotne jest sprzężenie systemu z procesem zarządzania fałszywymi alarmami stosowanym w środowisku docelowym:

- Szczególną uwagę należy zwrócić na przypadki wysokiej złożoności danych oraz szumu, gdzie systemy wykazują tendencję do fałszywych alarmów.
- Do rozważenie wprowadzenie mechanizmu wykorzystania kontekstu historycznego jako dodatkowy stopień walidacji alarmów - poprzez uczenie się z błędnych klasyfikacji.

5.3 Kierunki dalszego rozwoju

Przeprowadzone eksperymenty wskazują na kilka obiecujących kierunków dalszych badań. W trakcie prac zidentyfikowano następujące zagadnienia do badań i rozwoju:

- Zbadanie zdolności metody do wykrywania zjawiska concept driftu, oraz sygnalizowania stopnia prawdopodobieństwa tego zjawiska - do decyzji i analizy przez oparatorów ludzkich. W dalszych krokach użyteczne praktycznie byłoby połączenie wykrycia podejrzenia o concept drift z automatyzacją przeuczania modelu.

- Wdrożenie i zbadanie skuteczności wdrożenia różnych mechanizmów opartych o uczenie częściowo nadzorowane. W kontekście prac wdrożeniowych wprowadzenie tego aspektu będzie ułatwione - partner przemysłowy rozwija aktywnie zestaw narzędzi wspierających ocenę istotności poszczególnych punktów danych pod kątem ich wpływu na jakość klasyfikatora - bibliotekę Labeling-in-the-Loop [45].

- Zweryfikowanie metody na potrzeby diagnostyki modeli przetwarzających inne typy danych - w szczególności szeregi czasowe.

- Zweryfikowanie metody na potrzeby diagnostyki modeli przetwarzających dane wielomodalne.

- Zweryfikowanie możliwości rozszerzenia zestawu atrybutów diagnostycznych w celu umożliwienia skutecznej izolacji typów ataków.

- Zbadanie skuteczności metody dla bardziej złożonych architektur sieci neuronowych, w tym sieci konwolucyjnych i rekurencyjnych.

- Analiza wpływu struktury danych na skuteczność detekcji, szczególnie w kontekście różnic między danymi tabelarycznymi a innymi typami danych (obrazy, tekst).

- Wprowadzenie uzupełniających mechanizmów monitorowania zmian w lokalnej strukturze danych, jako uzupełnienie atrybutów diagnostycznych.

- Optymalizacja i automatyzacja procesu aktualizacji klasyfikatorów w odpowiedzi na nowe typy ataków.

Dalsze badania będą prowadzone przez autora oraz zespół badawczo-wdrożeniowy partnera w ramach prowadzonych niezależnie prac, w tym z wykorzystaniem rzeczywistych danych i przykładów klientów.

Podziękowania

Chciałbym podziękować wszystkim osobom, bez których ta praca doktorska nie tylko by nie powstała, ale też nie byłaby tak fascynującym doświadczeniem.

Przede wszystkim dziękuję mojemu promotorowi, dr. hab. Markowi Sikorze, za cierpliwość, zrozumienie i umiejętność przekształcania moich chaotycznych czasem pomysłów w spójną całość. Dziękuję za wszystkie cenne uwagi do pracy, które znacząco wpłynęły na jej ostateczny kształt. I jeszcze raz za cierpliwość.

Dr. Łukaszowi Wawrowskiemu dziękuję za owocną współpracę naukową i wspólne publikacje. Nasze długie dyskusje, wykraczające często poza bezpośredni obszar badań, były źródłem wielu cennych inspiracji i pozwoliły spojrzeć na problemy badawcze z nowej perspektywy.

Prof. Dominikowi Ślęzakowi oraz Jackowi Puczniewskiemu jestem wdzięczny za umożliwienie realizacji pracy i stworzenie w QED Software środowiska, gdzie można skutecznie łączyć pracę zawodową z działalnością naukową. Dziękuję za zaufanie i elastyczność, które umożliwiły mi realizację badań.

Całemu zespołowi QED Software dziękuję za wspaniałą atmosferę pracy i nieocenione wsparcie. Szczególne podziękowania za cierpliwe znoszenie moich opowieści o badaniach podczas przerw kawowych - obiecuję, że teraz z kolei ja będę częściej słuchał o Waszych projektach.

Szczególne podziękowania kieruję do mojej najbliższej rodziny - Agnieszki, Doroty i Witolda - za nieustające wsparcie, cierpliwość i wyrozumiałość. Dziękuję za wszystkie chwile, gdy musieliście dzielić mnie z pracą naukową, za dodawanie otuchy w momentach zwątpienia i za zainteresowanie (nawet jeśli miejscami lekko naciągane) czym właściwie się zajmuję. Bez Waszego wsparcia i niekończących się kubków herbaty ta droga byłaby znacznie trudniejsza.

Na koniec dziękuję wszystkim, którzy w jakikolwiek sposób przyczynili się do powstania tej pracy, również choćby pytaniami ”kiedy w końcu bronisz ten doktorat?” - wasza uporczywość była dodatkową motywacją do ukończenia tej pracy.

Bibliography

[1] Abad-Rocamora, E., Wu, Y., Liu, F., Chrysos, G., Cevher, V. Revisiting character-level adversarial attacks for language models. *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024.

[2] Alzantot, M., Sharma, Y., Elgohary, A., Ho, B.-J., Srivastava, M., Chang, K.-W. Genattack: Practical black-box attacks with gradient-free optimization. *arXiv preprint arXiv:1805.11090*, 2019.

[3] Amodei, D., Olah, C., Steinhardt, J., Christiano, P. F., Schulman, J., Mané, D. Concrete problems in ai safety. *ArXiv*, abs/1606.06565, 2016.

[4] Anelli, V. W., Bellogín, A., Deldjoo, Y., Di Noia, T., Merra, F. A. Multi-step adversarial perturbations on recommender systems embeddings. *arXiv preprint arXiv:2010.01329*, 2020.

[5] Athalye, A., Engstrom, L., Ilyas, A., Kwok, K. Synthesizing robust adversarial examples. *International Conference on Machine Learning*, strony 284–293. PMLR, 2018.

[6] Ballet, V., Renard, X., Aigrain, J., Laugel, T., Frossard, P., Detyniecki, M. Adversarial attacks on tabular data: A survey. *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2019.

[7] Barreno, M., Nelson, B., Joseph, A. D., Tygar, J. D. The security of machine learning. *Machine Learning*, 81(2):121–148, 2010.

[8] Ben-Haim, A., Bronstein, M. M., Marom, E. Imperceptible adversarial attacks on tabular data. *arXiv preprint arXiv:1911.03274*, 2019.

[9] Biggio, B., Fumera, G., Roli, F. Security evaluation of pattern classifiers under attack. *IEEE Transactions on Knowledge and Data Engineering*, 26(4):984–996, 2014.

[10] Biggio, B., Roli, F. Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recognition*, 84:317–331, 2018.

[11] Borisov, V., Leemann, T., Seßler, K., Haug, J., Pawelczyk, M., Kasneci, G. Deep neural networks and tabular data: A survey. *arXiv preprint arXiv:2110.01889*, 2021.

[12] Breiman, L. Random forests. *Machine learning*, 45:5–32, 2001.

[13] Brendel, W., Rauber, J., Bethge, M. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. *ArXiv*, abs/1712.04248, 2017.

[14] Carlini, N., Wagner, D. Towards evaluating the robustness of neural networks. strony 39–57, 05 2017.

[15] Cartella, F., Anunciação, O., Funabiki, Y., Yamaguchi, D., Akishita, T., Elshocht, O. Adversarial attacks for tabular data: Application to fraud detection and imbalanced data. 01 2021.

[16] Chakraborty, A., Alam, M., Dey, V., Chattopadhyay, A., Mukhopadhyay, D. A survey on adversarial attacks and defences. *CAAI Transactions on Intelligence Technology*, 6(1):25–45, 2021.

[17] Chen, J., Jordan, M. I., Wainwright, M. J. Hopskipjumpattack: A query-efficient decision-based attack. *arXiv preprint arXiv:1904.02144*, 2019.

[18] Chen, P.-Y., Zhang, H., Sharma, Y., Yi, J., Hsieh, C.-J. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, strony 15–26, 2017.

[19] Demontis, A., Melis, M., Biggio, B., Maiorca, D., Arp, D., Rieck, K., Roli, F. Yes, machine learning can be more secure! a case study on android malware detection. *IEEE Transactions on Dependable and Secure Computing*, 16(4):711–724, 2019.

[20] Dougherty, J., Kohavi, R., Sahami, M. Supervised and unsupervised discretization of continuous features. *Machine learning proceedings 1995*, strony 194–202. Elsevier, 1995.

[21] Ebrahimi, J., Rao, A., Lowd, D., Dou, D. HotFlip: White-box adversarial examples for text classification. Gurevych, I., Miyao, Y., redaktorzy, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, strony 31–36, Melbourne, Australia, Lip. 2018. Association for Computational Linguistics.

[22] Evtimov, I., Eykholt, K., Fernandes, E., Kohno, T., Li, B., Prakash, A., Rahmati, A., Song, D. Robust physical-world attacks on machine learning models. *CoRR*, abs/1707.08945, 2017.

[23] Finlayson, S. G., Bowers, J. D., Ito, J., Zittrain, J. L., Beam, A. L., Kohane, I. S. Adversarial attacks on medical machine learning. *Science*, 363(6433):1287–1289, 2019.

[24] Frederickson, C., Moore, M., Dawson, G., Polikar, R. Attack strength vs. detectability dilemma in adversarial machine learning. *2018 international joint conference on neural networks (IJCNN)*, strony 1–8. IEEE, 2018.

[25] Fu, J., Sun, J., Wang, G. Boosting black-box adversarial attacks with meta learning. *2022 41st Chinese Control Conference (CCC)*, strony 7308–7313, 2022.

[26] Gong, Z., Wang, W. Adversarial and clean data are not twins. strony 1–5, 06 2023.

[27] Goodfellow, I. J., Shlens, J., Szegedy, C. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[28] Grosse, K., Papernot, N., Manoharan, P., Backes, M., McDaniel, P. Adversarial perturbations against deep neural networks for malware classification. 06 2016.

[29] Grosse, K., Papernot, N., Manoharan, P., Backes, M., McDaniel, P. Statistical detection of adversarial examples. *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2017.

[30] Gu, S., Rigazio, L. Towards deep neural network architectures robust to adversarial examples. *arXiv preprint arXiv:1412.5068*, 2014.

[31] Hall, P. On the Art and Science of Machine Learning Explanations, 2018.

[32] Hashemi, M., Fathi, A. Permuteattack: Counterfactual explanation of machine learning credit scorecards, 2020.

[33] He, K., Zhang, X., Ren, S., Sun, J. Deep residual learning for image recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, strony 770–778, 2016.

[34] He, Z., Ouyang, C., Alzubaidi, L., Barros, A., Moreira, C. Investigating imperceptibility of adversarial attacks on tabular data: An empirical analysis. *Intelligent Systems with Applications*, 25:200461, 2025.

[35] Hendrycks, D., Gimpel, K. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

[36] Hornik, K., Stinchcombe, M., White, H. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2:359–366, 1989.

[37] Huang, S., Papernot, N., Goodfellow, I., Duan, Y., Abbeel, P. Adversarial attacks on neural network policies. 02 2017.

[38] Ilyas, A., Engstrom, L., Athalye, A., Lin, J. Black-box adversarial attacks with limited queries and information. *International Conference on Machine Learning*, 2018.

[39] Jagielski, M., Oprea, A., Biggio, B., Liu, C., Nita-Rotaru, C., Li, B. Manipulating machine learning: Poisoning attacks and countermeasures for regression learning. *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*, strony 19–35, 2018.

[40] Janusz, A., Ślęzak, D. Random Probes in Computation and Assessment of Approximate Reducts. Kryszkiewicz, M., Cornelis, C., Ciucci, D., Medina-Moreno, J., Motoda, H., Raś, Z. W., redaktorzy, *Rough Sets and Intelligent Systems Paradigms - Second International Conference, RSEISP 2014, Held as Part of JRS 2014, Granada and Madrid, Spain, July 9-13, 2014. Proceedings*, wolumen 8537 serii *Lecture Notes in Computer Science*, strony 53–64. Springer, 2014.

[41] Janusz, A., Zalewska, A., Wawrowski, L., Biczyk, P., Ludziejewski, J., Sikora, M., Ślęzak, D. Brightbox—a rough set based technology for diagnosing mistakes of machine learning models. *Applied Soft Computing*, strona 110285, 2023.

[42] Jiang, F., Yu, X., Du, J., Gong, D., Zhang, Y., Peng, Y. Ensemble Learning Based on Approximate Reducts and Bootstrap Sampling. *Information Sciences*, 547:797–813, 2021.

[43] Kang, D., Sun, Y., Hendrycks, D., Brown, T. B., Steinhardt, J. Testing robustness against unforeseen adversaries. *ArXiv*, abs/1908.08016, 2019.

[44] Kantchelian, A., Tygar, J. D., Joseph, A. D. Evasion and hardening of tree ensemble classifiers. *Proceedings of the 2016 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, strony 1135–1144, 2016.

[45] Kałuża, D., Janusz, A., Ślęzak, D. Robust assignment of labels for active learning with sparse and noisy annotations. *ECAI 2023*, 09 2023.

[46] Kireev, K., Kulynych, B., Troncoso, C. Adversarial robustness for tabular data through cost and utility awareness. *NeurIPS ML Safety Workshop*, 2022.

[47] Kurakin, A., Goodfellow, I., Bengio, S. Adversarial examples in the physical world. 07 2016.

[48] Lindsay, R. K., Buchanan, B. G., Feigenbaum, E. A., Lederberg, J. Dendral: a case study of the first expert system for scientific hypothesis formation. *Artificial intelligence*, 61(2):209–261, 1993.

[49] Liu, C., Salzmann, M., Lin, T., Tomioka, R., Süsstrunk, S. On the loss landscape of adversarial training: Identifying challenges and how to overcome them. *Advances in Neural Information Processing Systems*, 2020.

[50] Liu, F. T., Ting, K. M., Zhou, Z.-H. Isolation forest. *IEEE International Conference on Data Mining (ICDM)*, strony 413–422, 2008.

[51] Ma, X., Li, B., Wang, Y., Erfani, S. M., Wijewickrema, S., Schoenebeck, G., Houle, M. E., Bailey, J., Chi, L. Characterizing adversarial subspaces using local intrinsic dimensionality. *International Conference on Learning Representations (ICLR)*, 2018.

[52] Madry, A., Makelov, A., Schmidt, L., Tsipras, D., Vladu, A. Towards deep learning models resistant to adversarial attacks. *International Conference on Learning Representations*, 2018.

[53] Mafarja, M. M., Mirjalili, S. Hybrid Binary Ant Lion Optimizer with Rough Set and Approximate Entropy Reducts for Feature Selection. *Soft Computing*, 23(15):6249–6265, 2019.

[54] McCane, B., Albert, M. Distance functions for categorical and mixed variables. *Pattern Recognition Letters*, 29(7):986–993, 2008.

[55] Montufar, G. F., Pascanu, R., Cho, K., Bengio, Y. On the number of linear regions of deep neural networks. *Advances in Neural Information Processing Systems*, wolumen 27, 2014.

[56] Moosavi-Dezfooli, S.-M., Fawzi, A., Frossard, P. Deepfool: A simple and accurate method to fool deep neural networks. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, strony 2574–2582, 2016.

[57] National Institute of Standards and Technology. Adversarial machine learning: A taxonomy and terminology of attacks and mitigations. Raport instytutowy, NIST, 2023.

[58] Nicolae, M.-I., Sinn, M., Tran, M. N., Buesser, B., Rawat, A., Wistuba, M., Zantedeschi, V., Baracaldo, N., Chen, B., Ludwig, H., i in. Adversarial robustness toolbox v1. 0.0. *arXiv preprint arXiv:1807.01069*, 2018.

[59] Papernot, N., McDaniel, P. Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning. 03 2018.

[60] Papernot, N., McDaniel, P., Goodfellow, I. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. 05 2016.

[61] Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z. B., Swami, A. Practical black-box attacks against deep learning systems using adversarial examples. *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, strony 506–519, 2017.

[62] Pawlak, Z. *Rough sets: Theoretical aspects of reasoning about data*. Springer Science & Business Media, 1991.

[63] Pedraza, A., Deniz, O., Singh, H., Bueno, G. Leveraging autoencoders and chaos theory to improve adversarial example detection. *Neural Computing and Applications*, 36:18265–18275, 07 2024.

[64] Pierazzi, F., Pendlebury, F., Cortellazzi, J., Cavallaro, L. Intriguing properties of adversarial ml attacks in the problem space. *2020 IEEE Symposium on Security and Privacy (SP)*, strony 1332–1349, 2020.

[65] Riza, L. S., Janusz, A., Bergmeir, C., Cornelis, C., Herrera, F., Ślęzak, D., Benítez, J. M. Implementing Algorithms of Rough Set Theory and Fuzzy Rough Set Theory in the R Package "RoughSets". *Information Sciences*, 287:68–89, 2014.

[66] Rosenberg, I., Shabtai, A., Rokach, L. Generic black-box end-to-end attack against state-of-the-art api call based malware classifiers. *Proceedings of the ACM Asia Conference on Computer and Communications Security (AsiaCCS)*, strony 490–500, 2018.

[67] Rudin, C. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, 2019.

[68] Sakurada, M., Yairi, T. Anomaly detection using autoencoders with nonlinear dimensionality reduction. *Proceedings of the MLSDA Workshop on Machine Learning for Sensory Data Analysis*, strony 4–11, 2014.

[69] Sethi, T., Kantardzic, M. 'security theater': On the vulnerability of classifiers to exploratory attacks. 05 2017.

[70] Shavitt, I., Segal, E. Regularization learning in shallow and deep networks for tabular data. *Advances in Neural Information Processing Systems*, 31, 2018.

[71] Somepalli, G., Goldblum, M., Schwarzschild, A., Gupta, A., Goldstein, T. Well-tuned simple nets excel on tabular datasets. *Advances in Neural Information Processing Systems*, wolumen 34, strony 23261–23272, 2021.

[72] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[73] Stawicki, S., Ślęzak, D., Janusz, A., Widz, S. Decision Bireducts and Decision Reducts – A Comparison. *International Journal of Approximate Reasoning*, 84:75–109, 2017.

[74] Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., Fergus, R. Intriguing properties of neural networks. *International Conference on Learning Representations*, 2013.

[75] Uesato, J., O'Donoghue, B., Kohli, P., van den Oord, A. Adversarial risk and the dangers of evaluating against weak attacks. Dy, J., Krause, A., redaktorzy, *Proceedings of the 35th International Conference on Machine Learning*, wolumen 80 serii *Proceedings of Machine Learning Research*, strony 5025–5034. PMLR, 10–15 Jul 2018.

[76] Wang, G., Yan, H., Guo, Y., Wei, X. Improving adversarial transferability with gradient refining. *arXiv preprint arXiv:2105.04834*, 2021.

[77] Wang, Z., Wang, W., Chen, Q., Wang, Q., Nguyen, A. Generating valid and natural adversarial examples with large language models. strony 1716–1721, 05 2024.

[78] Xie, C., Zhang, Z., Zhou, Y., Bai, S., Wang, J., Wang, Z., Yuille, A. L. Improving transferability of adversarial examples with input diversity. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, strony 2730–2739, 2019.

[79] Xu, W., Evans, D., Qi, Y. Feature squeezing: Detecting adversarial examples in deep neural networks. 01 2018.

[80] Yin, F., Zhang, Y., Wu, B., Feng, Y., Zhang, J., Fan, Y., Yang, Y. Generalizable Black-Box Adversarial Attack With Meta Learning . *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 46(03):1804–1818, Mar. 2024.

[81] Yuan, Z., Zhang, J., Shan, S. Adaptive perturbation for adversarial attack. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46:5663–5676, 2021.

[82] Zhang, C., Zhang, H., Hsieh, C.-J. An efficient adversarial attack for tree ensembles. *Proceedings of the 34th International Conference on Neural Information*

Processing Systems, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc.

[83] Zheng, M., Yan, X., Zhu, Z., Chen, H., Wu, B. Blackboxbench: A comprehensive benchmark of black-box adversarial attacks. *ArXiv*, abs/2312.16979, 2023.

[84] Zhu, X., Dong, J., qi, j., Zhou, Z., Dong, Z., Sun, Y., Wang, M. Auth: An adversarial autoencoder based unsupervised insider threat detection scheme for multisource logs. *IEEE Transactions on Industrial Informatics*, PP:1–12, 09 2024.