

POLITECHNIKA WROCŁAWSKA

WYDZIAŁ ELEKTRONIKI

KIERUNEK: INFORMATYKA

SPECJALNOŚĆ: SYSTEMY INFORMATYKI W MEDYCYNIE

PRACA DYPLOMOWA

MAGISTERSKA

Analiza porównawcza wybranych metod
selekcji cech w zadaniu klasyfikacji danych
niezbalansowanych.

Comparative analysis of feature selection
techniques in imbalanced data classification
problems.

AUTOR:

Maciej Hajduk

PROWADZĄCY PRACĘ:

dr inż. Paweł Trajdos, K02W04D03

Spis treści

1	Wstęp	6
1.1	Wprowadzenie	6
2	Analiza problemu	9
2.1	Przegląd literatury	9
2.2	Cel selekcji cech	10
2.3	Podstawowy podział	11
2.3.1	Metody rankingowe	11
2.3.2	Metody opakowane	12
2.3.3	Metody wbudowane	12
2.4	Selekcja cech a ekstrakcja	12
2.5	Problem niezrównoważonego rozkładu klas	12
2.6	Metody klasyfikacji danych niezbalansowanych	14
2.6.1	Metody na poziomie danych	14
2.6.2	Metody na poziomie algorytmów	14
2.6.3	Podejścia hybrydowe	15
2.7	Metody oparte o selekcję cech	15
2.7.1	Correlation coefficient	16
2.7.2	Chi-square	17
2.7.3	Information Gain	18
2.7.4	Relief i ReliefF	19
2.7.5	ANOVA	20
3	Założenia i plan eksperymentu	23
3.1	Generowanie wyników	23
3.2	Ocena działania algorytmów	24
3.3	Generowanie zbioru treningowego i testowego	24
3.4	Zbiory danych	25
3.4.1	Credit Card Fraud Detection	25
3.4.2	Health Insurance Cross Sell	26
3.4.3	Mushroom Classification	27
3.4.4	Zbiór „Custom”	28
3.4.5	Inne zbiory danych	28
3.5	Przygotowanie danych	28
3.5.1	Preprocessing zbioru CCFD	29
3.5.2	Preprocessing zbioru HICS	30
3.5.3	Preprocessing zbioru Mushroom	30
3.5.4	Preprocessing zbioru Custom	30
3.5.5	Preprocessing pozostałych zbiorów	30
3.6	Eksperymenty	30

4 Wyniki	33
5 Wnioski	35
6 Bibliografia	37
Zawartość płyty CD	42

Spis rysunków

2.1	Trzy główne podejścia do problemu selekcji cech	11
2.2	Przykład niezrównoważonego rozkładu klas	13
2.3	Interpretacje współczynnika korelacji dla różnych zbiorów danych	17
2.4	Wybór elementów <i>near hit</i> oraz <i>near miss</i> w każdej iteracji działania algorytmu	20
3.1	Graficzna reprezentacja działania algorytmu KNN	23
3.2	Graficzna reprezentacja działania algorytmu walidacji krzyżowej	25
3.3	Dystrybucja klas i danych dla zbioru Credit Card Fraud Detection	26
3.4	Dystrybucja klas i danych dla zbioru Health Insurance Cross Sell	27
3.5	Dystrybucja klas i danych dla zbioru Mushroom	27
3.6	Dystrybucja klas i danych dla zbioru Custom	28
3.7	Dystrybucja wartości dla cech <i>Time</i> oraz <i>Amount</i>	29

Spis tablic

1 Wstęp

1.1 Wprowadzenie

Celem pracy jest porównanie różnych metod selekcji cech w problemie trenowania algorytmów uczenia maszynowego na danych niezbalansowanych. W jej ramach, przedstawione i opisane zostaną popularne obecnie metody selekcji oraz przeprowadzone zostaną eksperymenty dla przykładowych zbiorów danych, zarówno rzeczywistych jak i syntetycznych, celem których będzie stworzenie rankingu algorytmów. Autor sprawdził, jak właściwie przeprowadzona selekcja wpływa na jakość wyników dostarczanych przez klasyfikator i jak przytoczone przez niego metody radzą sobie z danymi, w których występuje znaczna przewaga liczebności jednej bądź kilku klas. Aspekt inżynierski polegał na implementacji zaproponowanych w pracy eksperymentów, co pozwoliło na kompleksowe porównanie algorytmów.

The aim of this paper is to compare different feature selection methods in the problem of training machine learning algorithms on unbalanced data. In it, the currently popular selection methods will be presented and described, and experiments will be conducted for example datasets, both real and synthetic, with the goal of creation the ranking of algorithms. Author checked how properly performed selection affects the quality of results provided by the classifier and how the methods quoted by him deal with data in which there is a significant predominance of one or more or a few classes. The engineering aspect consisted in the implementation of the experiments proposed in the paper, which allowed for a comprehensive comparison of the algorithms.

Praca swoim zakresem objęła porównanie popularnych metod selekcji cech w ramach kilku wybranych zbiorów danych. Napisany w jej ramach program pozwala na kompleksowe porównanie wyników różnych algorytmów przetestowanych na kilku zestawach danych. Dane te posłużyć mogą do wyboru najodpowiedniejszej metody we wszelkich problemach klasyfikacyjnych, w których elementy wykazują szczególną nadreprezentację jednej bądź kilku klas. Zawarte w pracy podsumowanie zawrze wyniki przeprowadzonych przez autora badań.

Praca składa się z czterech rozdziałów:

Rozdział pierwszy: Na rozdział pierwszy składają się omówienie analizy wybranego problemu, przedstawienie motywacji podjęcia tego tematu oraz przegląd literatury. Zostały opisane w nim również podstawowe metody selekcji cech i wyjaśnienie różnicy pomiędzy selekcją oraz ekstrakcją. Szczegółowa charakterystyka zagadnienia opisana w rozdziale zawiera opis problemu jakim jest niezerównoważony rozkład klas w algorytmie uczenia maszynowego. Rozdział zawiera też szczegółowy opis poszczególnych, wykorzystanych później metod selekcji, wraz z ich matematyczną interpretacją.

Rozdział drugi: Założenia i plan eksperymentu. W rozdziale drugim zawarte zostaną informacje

związane z inżynierskim aspektem pracy, czyli projekt systemu, plan poszczególnych eksperymentów i opis danych, jakie użyte zostaną podczas doświadczeń. Swoim zakresem rozdział obejmie krótki opis użytych przez autora bibliotek oraz wykorzystywanych funkcji. Znajdą się tutaj również instrukcje instalacji i wdrożenia systemu dla potencjalnych środowisk docelowych.

Rozdział trzeci: Rozdział zawiera podsumowanie uzyskanych wyników oraz przedstawienie ich w czytelny i zrozumiały sposób.

Rozdział czwarty: W czwartym piątym zawarta zostanie interpretacja oraz konfrontacja wyników z hipotezą postawioną na początku pracy. Przedstawione zostaną ewentualne możliwości rozwoju projektu.

Udało się zrealizować wszystkie postawione cele.

2 Analiza problemu

Uczenie maszynowe to bardzo dynamicznie rozwijająca się gałąź informatyki. Niezwykła ekspansja wynika z zapotrzebowania na wykrywanie prawidłowości, uogólnianie oraz precyzowanie danych. Takie możliwości pozwoliły znaleźć zastosowanie dla algorytmów sztucznej inteligencji w bardzo wielu różnych branżach - począwszy od medycyny, poprzez finanse, produkcję i branżę rozrywkową. Tak duży przekrój różnych zastosowań wymaga ciągłego ulepszania istniejących już wzorców oraz wymyślania nowych, lepszych i bardziej efektywnych algorytmów. W większości praktycznych problemów do klasyfikacji obiektów, autor programista na dużej liczbie cech. Warto jednak pamiętać, że w tym przypadku wiele, nie oznacza lepszych rezultatów. Należy przytoczyć pojęcie „przekleństwa wielowymiarowości”[9]. Oznacza ono, że większy wymiar wymaga od programisty znacznie większej liczby danych, oraz wraz ze wzrostem liczby cech wykładniczo rośnie liczba możliwych wariantów dopasowań, co znacznie zwiększa złożoność obliczeniową naszych algorytmów.

Aby uniknąć problemów generowanych przez zbyt dużą ilość cech, a jednocześnie wykorzystać cechy, które zapewniają jak najlepszą separowalność klas, zazwyczaj pierwszym krokiem w zadaniu klasyfikacji jest selekcja lub ekstrakcja najodpowiedniejszych cech [17].

2.1 Przegląd literatury

Zarówno sam problem selekcji cech jak i sposoby radzenia sobie z nierównomiernym rozkładem klas to - szczególnie w ostatnich latach - często poruszany problem, co skutkuje dużym przekrojem prac, również w ujęciu czysto dziedzinowym - jak wykorzystanie konkretnych algorytmów dla bardzo konkretnych zastosowań.

Wstępną analizę problemu przedstawił Jakub Piątkowski w pracy *Analiza i rozwój metod doboru cech dla dużych problemów klasyfikacyjnych*. Autor wymienia istniejące metody doboru cech i szczegółowo je opisuje [7]. Jest to dobre wprowadzenie do problematyki klasyfikacji i selekcji cech, a każda przytoczona metoda ma również podłoże matematyczne opisane odpowiednimi wzorami. W podobnej pracy *Redukcja wymiarowości i selekcja cech w zadaniach klasyfikacji i regresji z wykorzystaniem uczenia maszynowego*, twórca zacytował wyniki swoich eksperymentów, które pozwoliły mu na tworzenie rankingów algorytmów [10]. Szczegółowo do problemu podszedł mgr Wiesław Chmielnicki, w swojej rozprawie: *Efektywne metody selekcji cech i rozwiązywania problemu wieloklasowego w nadzorowanej klasyfikacji danych*. Oprócz opisu znanych metod, artykuł zawiera również sugestie dotyczące nowych algorytmów hybrydowych, które w niektórych przypadkach dają lepsze wyniki niż metody tradycyjne [17]. Mark A. Hall porusza interesujący temat w swojej książce *Correlation-based Feature Selection for Machine Learning*, badając wartość zestawu cech na podstawie korelacji pomiędzy nimi. Autor przeprowadził szereg eksperymentów, porównał swoją metodę z metodami powszechnie stosowanymi, starając się między innymi wyodrębnić problemy, dla których jego algorytm jest najbardziej skuteczny [3].

Twórcy pracy *A Survey on Evolutionary Computation Approaches to Feature Selection* zajęli się przeglądem znanych metod tworząc dokument podsumowujący każdą z nich, z jej wadami oraz zaletami. Artykuł jest oparty na przeglądzie najnowszych prac w zadanej dziedzinie i pozwala na dobranie odpowiedniej metody do zadanego zadania [2]. Ważna dla tematu tej pracy jest również rozprawa *Zastosowanie wybranych metod przekształcania i selekcji danych oraz konstrukcji cech w zadaniach klasyfikacji i klasteryzacji* mgr inż. Piotra Płońskiego, podsumowująca niejako cały proces uczenia maszynowego i roli, jaką pełni selekcja cech w kontekście dobrze działającego systemu analizy danych [6]. W 2008 roku Chen Xuwen i Michael Wasikowski zaproponowali metodę zwaną FAST, która opiera się na powierzchni pod krzywą ROC, która jest generowana przez równomierne rozłożenie progów granicy decyzyjnej klasyfikatora pojedynczej cechy [11]. W 2014 roku D. Tiwari opisał algorytm wyboru cech dla niezerównoważonych zbiorów danych modyfikując popularny algorytm RELIEFF, aby rozwiązać problem nierównowagi klas [12]. W przypadku klas mniejszościowych metoda ta nadaje większą wagę atrybutom, co skutkuje wyższą wagą tych klas podczas samej klasyfikacji. Inna analiza CoIL Challenge 2000 przeprowadzona przez Elkana wykazała, że zwykłe algorytmy selekcji cech nie były wystarczająco dobre do zadania klasyfikacji danych niezbalansowanych [13]. Na etapie selekcji należało rozważyć interakcję między różnymi cechami. Największą wadą, jaką znalazł w przypadku większości stosowanych metod selekcji cech, jest to, że nie rozważali oni wybierania wysoce skorelowanych cech, ponieważ uważano je za zbędne. Guyon i Elisseeff przeprowadzili solidną analizę teoretyczną. Wykazali, że same nieistotne cechy mogą być przydatne w połączeniu z innymi cechami, a połączenie dwóch silnie skorelowanych cech może być lepsze niż każda z nich niezależnie [14].

2.2 Cel selekcji cech

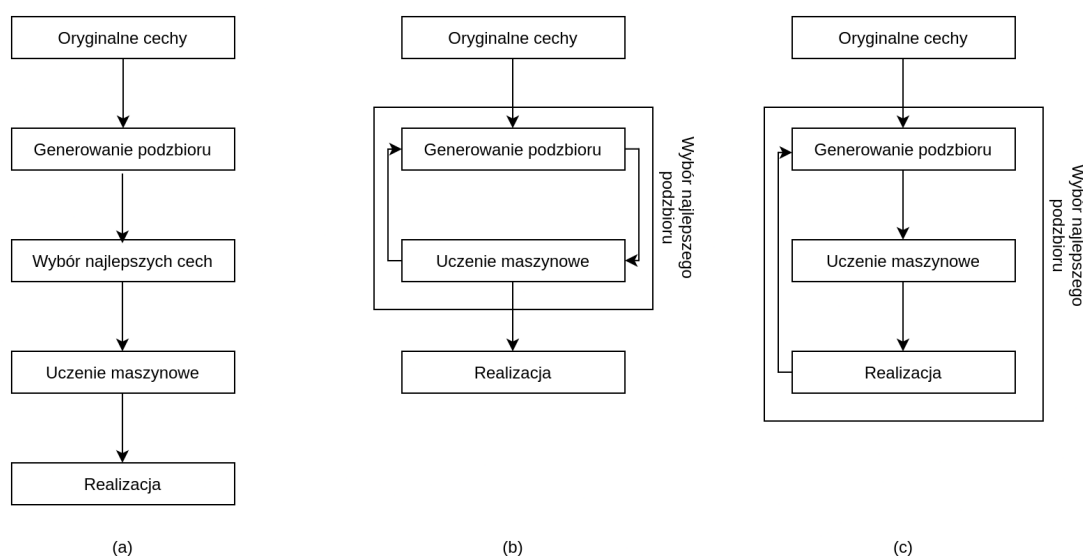
Selekcja cech polega na identyfikacji tych elementów puli cech, które uznawane są za najlepsze deskryptory rozważanych kategorii. Zaletą selekcji jest możliwość zbadania tych deskryptorów, które są istotne z punktu widzenia danego zadania klasyfikacji, czyli jednocześnie zrozumienia różnic między analizowanymi kategoriami. Poprzez proces ekstrakcji cech tracimy niestety bezpowrotnie część początkowych cech. Z tego powodu nie ekstrakcja, a selekcja cech jest obecnie najpowszechniejszą strategią służącą przygotowaniu reprezentacji analizowanych danych [17].

Selekcja cech jest odpowiedzialna za wybór najbardziej istotnych atrybutów badanych obiektów, co przekłada się bezpośrednio na poprawne działanie klasyfikatora. Dyspozycja coraz większymi bazami danych zmusza do optymalizacji tego procesu. Gwałtownie rosnąca liczba cech stanowi poważny problem - powoduje nie tylko wydłużenie procesu uczenia oraz wzrost złożoności klasyfikatora, ale niesie ze sobą także ryzyko spadku prawdopodobieństwa poprawnej klasyfikacji. Związane jest to z tak zwanym "przekleństwem wymiarowości" [15]. Zjawisko to zachodzi, gdy liczba cech znacznie przewyższa liczebność samego zbioru danych. Zadaniem selekcji cech jest również lepsze zrozumienie problemu oraz zmniejszenie kosztów archiwizacji przyszłych danych. W kolejnych rozdziałach opisane zostaną trzy główne metody tworzenia algorytmów selekcji: metody rankingowe - zwane filtrami, metody opakowane oraz metody wbudowane. Dla każdej z wymienionych metod zostanie określona idea, oraz przedstawione zostaną algorytmy

reprezentujące daną metodologię.

2.3 Podstawowy podział

W kolejnych rozdziałach opisane zostaną trzy główne metody tworzenia algorytmów selekcji: metody rankingowe - zwane filtrami, metody opakowane oraz metody wbudowane [34]. Dla każdej z wymienionych metod zostanie określona idea, oraz przedstawione zostaną algorytmy reprezentujące daną metodologię [7, 10, 3].



Rysunek 2.1: Trzy główne podejścia do problemu selekcji cech

2.3.1 Metody rankingowe

Najprostsze podejście do problemu selekcji cech reprezentowane jest właśnie poprzez metody rankingowe, nazywane też filtrami [34]. Jak sama nazwa wskazuje do zadania selekcji przy pomocy metod rankingowych podchodzimy wyróżniając w zbiorze cech następujące grupy: cechy istotne, nieistotne i redundantne. Istotne to takie cechy, które odróżniają od siebie klasy, nieistotne to takie, które nie niosą informacji dla problemu klasyfikacji, a cechy redundantne to takie których role z powodzeniem mogą przejąć inne cechy. Metody rankingowe polegają więc na znalezieniu pewnej miary pozwalającej stworzyć taki ranking cech, a potem wybrać najlepsze cechy, a odrzucić najgorsze. Metody rankingowe zazwyczaj są najszybsze i - co istotne - nie zależą one od używanej metody analizy danych [4, 17]. Ich istotną wadą stanowi brak możliwości uwzględnienia zależności pomiędzy cechami [16]. Kolejne opisane typy metod selekcji cech tej wady nie posiadają. Podejście rankingowe do selekcji cech zostało zobrazowane na grafice 2.1, w podpunkcie (a).

2.3.2 Metody opakowane

Podstawowymi metodami selekcji cech są metody opakowane, tak zwane wrappery [34]. W przeciwieństwie do metod rankingowych, w których selekcja cech i klasyfikator pozostają niezależne, w algorytmach opakowanych selekcji, ocena atrybutów dokonuje się przy użyciu konkretnego modelu. To właśnie efektywność samego klasyfikatora służy za miarę skuteczności metody. Zaletą tej metody jest jej uniwersalność i dokładność, natomiast wadą - wysoka złożoność obliczeniowa [34]. Dla efektywności tych algorytmów istotny jest sposób ustalania podzbioru cech [4, 17]. Wśród wielu metod wyszukiwania tegoż, wyróżnić można najprostszą - przeszukiwanie całego zbioru podzbiorów. Jest to jednak rozwiązanie bardzo kosztowne. Wobec tego typowymi strategiami są: przeszukiwanie w przód, przeszukiwanie wstecz oraz tworzenie indywidualnego rankingu [6]. Podejście opakowane do selekcji cech zostało zobrazowane na grafice 2.1, w podpunkcie (b).

2.3.3 Metody wbudowane

Metody wbudowane zawierają się w algorytmie klasyfikacji i to na etapie tworzenia modelu przypisuje się poszczególnym cechom wagi lub przeprowadza się ich eliminację. Do algorytmów klasyfikacji z wbudowaną metodą selekcji zaliczyć można popularne LASSO i RIDGE [2, 35, 36]. W literaturze natknąć się też można na podpięcie do tej kategorii metody wektorów nośnych (SVM) czy też analizy składowych głównych (PCA) [2]. Zaletą tych metod jest ich szybkość, ponieważ użycie ich nie wiąże się z dodatkowymi operacjami na zbiorze [2, 34, 36]. Podejście wbudowane do selekcji cech zostało zobrazowane na grafice 2.1, w podpunkcie (c).

2.4 Selekcja cech a ekstrakcja

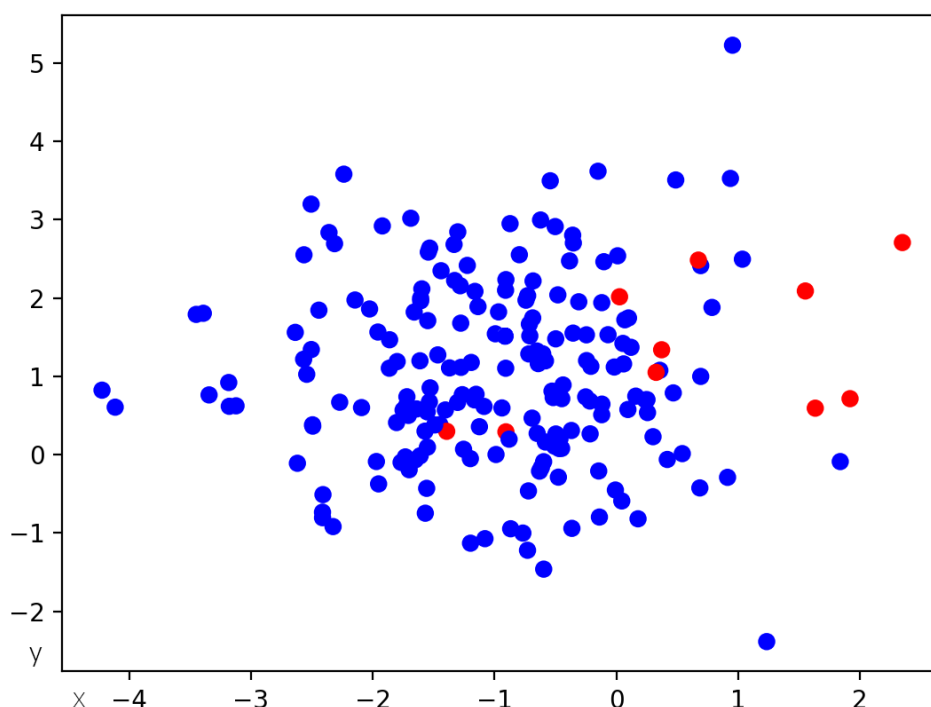
Selekcja cech ma na celu wybranie pewnych atrybutów opisujących dane pod kątem tego, czy nadają się one do dalszego wykorzystania w klasyfikacji, przy jednoczesnym odrzuceniu innych danych [37]. Zawsze rozważana jest ona w kontekście kolejnych zadań i nie można oceniać jej skuteczności w oderwaniu od wyników metody klasyfikacji wykorzystującej wybrane cechy. W większości przypadków budowany jest złożony model, który może zawierać jeden lub więcej algorytmów selekcji i co najmniej jeden klasyfikator. Ekstrakcja cech natomiast polega na utworzeniu nowego zestawu atrybutów poprzez liniową lub nieliniową kombinację oryginalnych danych. W przeciwieństwie do selekcji, gdzie celem jest zawsze uzyskanie podzbioru wszystkich atrybutów, wykorzystanie ekstrakcji wiąże się z wymiarem przestrzennym mniejszym, równy lub nawet większy od wymiaru przestrzeni startowej [37].

2.5 Problem niezrównoważonego rozkładu klas

Wśród wielu dobrze zbadanych i szeroko wykorzystywanych rozwiązań bazujących na uczeniu maszynowym, najbardziej obiecującymi są te, mające ratować ludzkie życie. Złożone choroby,

takie jak rak mózgu, stanowią poważne dla niego zagrożenie. Postęp w dziedzinie sztucznej inteligencji i metodach statystycznych stworzył nowe możliwości klasyfikacji i diagnozy najbardziej śmiertelnych chorób, takich jak rak, choroba Alzheimera, cukrzyca itp [17]. Z przypadkami takimi wiąże się jednak problem niezrównoważonej dystrybucji klas [38].

Niezrównoważony rozkład klas ma miejsce, gdy co najmniej jedna klasa jest niewystarczająco reprezentowana i przytłoczona przez inne klasy. Model klasyfikacji dla niezrównoważonych danych stwarza wiele przeszkód w uczeniu się algorytmów i przedstawia liczne konsekwencje dla rzeczywistych zastosowań. Ten problem powoduje niedoceniając przykładów klas mniejszościowych i powoduje niedokładne wyniki klasyfikacji w stosunku do przykładów klas większościowych. Klasyfikacja niezrównoważonego zbioru danych staje się trudniejsza przy ograniczonej liczbie próbek i ogromnej liczbie cech. Przykład takiego problemu zaobserwować można na grafice 2.2. Zawiera ona 200 elementów z których tylko 5% należy do klasy mniejszościowej - czerwonej.



Rysunek 2.2: Przykład niezrównoważonego rozkładu klas

Taka sytuacja jest problemem, ponieważ większość tradycyjnych algorytmów uczenia maszynowego trenowana na podobnym zbiorze, obciążona jest w stosunku do klasy bardziej licznej [19]. Jednocześnie, zazwyczaj lepsze zrozumienie klas mniej licznych jest istotniejsze z punktu widzenia problemu w ujęciu biznesowym [38]. Problemem jest również określenie jakości wyników algorytmu. Jakość klasyfikacji czyli dokładność - używana jako metryka ewaluacji może być w takim przypadku niewystarczająca, gdyż nawet model o skuteczności 95% - co jest na ogół

wartością bardzo dobrą - mógłby nie rozpoznawać żadnego elementu klasy mniejszościowej [19].

2.6 Metody klasyfikacji danych niebalansowanych

Problem nierównoważnego rozkładu przyciąga w ostatnim czasie zainteresowanie dużej części społeczności zajmującej się uczeniem maszynowym i eksploracją danych, zarówno ze środowisk akademickich jak i w przemyśle co znajduje odbicie w dużej liczbie statupów opierających swoje produkty i usługi na rozwiązaniach *machine-learningowych*. W ciągu kilkunastu ostatnich lat wyklarowały się trzy główne podejścia do uczenia modeli na danych niebalansowanych [20-23]. Są to metody na poziomie danych, metody na poziomie algorytmów oraz metody hybrydowe.

2.6.1 Metody na poziomie danych

Metody na poziomie danych (Data-level methods), modyfikują dostępne instancje problemu w celu jego zbalansowania. Można je dalej podzielić na podgrupy: metody próbkowania danych (data-sampling) i metody wyboru cech (feature selection methods) [18]. Metody nadpróbkowania i podpróbkowania stanowią dwie podgrupy metod próbkowania danych, w których próbkowanie danych z danego zbioru danych odbywa się losowo lub z wykorzystaniem określonego algorytmu. W procesie oversamplingu (nadpróbkowania) do danego zbioru danych dodawane są instancje klasy mniejszościowej (poprzez replikację), gdzie replikacja odbywa się losowo lub z wykorzystaniem algorytmów takich jak ADASYN [39]. W procesie undersamplingu natomiast, większość wystąpień klasy zostanie usuniętych z danego zbioru danych, a usuwanie odbywa się w dużej mierze losowo. SMOTE (Synthetic Minority Over-Sampling), to technika próbkowania polegająca na sztucznym ponownym próbkowaniu zbioru danych. Końcowym jej wynikiem jest zbiór danych o zrównoważonym rozkładzie [21]. Chociaż metody te mogą skutkować znacznie lepszymi wynikami w porównaniu z oryginalnym zestawem danych, istnieją poważne problemy związane z jej wykorzystaniem [9]. Po pobraniu zbyt małej liczby próbek wydajność klasyfikatora może ulec pogorszeniu z powodu potencjalnej utraty przydatnych przykładów klasy większości. Podobnie dodatkowe przypadki szkoleniowe wprowadzone przez nadmierne próbkowanie mogą zwiększyć złożoność obliczeniową klasyfikatora. W najgorszym przypadku dokładne kopie przykładów po nadmiernym próbkowaniu mogą prowadzić do nadmiernego dopasowania klasyfikatora [18]. Chociaż metody selekcji cech są powszechnie stosowane w celu poprawy wyników klasyfikacji, mogą one również pomóc w wyborze najbardziej wpływowych cech w celu wygenerowania unikalnej wiedzy do klasyfikacji. Zmniejsza to niekorzystny wpływ nierównowagi klas na wyniki klasyfikacji [18].

2.6.2 Metody na poziomie algorytmów

Metody na poziomie algorytmów (Algorithm-level methods), modyfikują istniejące algorytmy uczenia maszynowego. Można je podzielić na metody wrażliwe na koszty (cost-sensitive me-

thods) i metody zintegrowane [18]. Pierwsza z nich opiera się na zasadzie przypisywaniu większej wagi instancjom w przypadku błędnej klasyfikacji. Na przykład fałszywie negatywnym przewidywaniom można przypisać wyższy koszt niż fałszywie dodatnim przewidywaniom. Metody zintegrowane mogą być również stosowane jako metody wrażliwe na koszty, w przypadku których wynikiem klasyfikacji jest pewna kombinacja wielu klasyfikatorów zbudowanych na zbiorze danych [18]. Bagging i Boosting to dwa powszechne typy metod uczenia zintegrowanego [40]. Bagging minimalizuje wariancję, generując kilka zestawów uczących z danego zestawu danych i generując klasyfikator dla każdego zestawu uczącego, a następnie łącząc ich modele w celu ostatecznej klasyfikacji. Algorytmy wykorzystujące Boosting, podobne do algorytmu AdaBoost, tworzą serię klasyfikatorów, wszystkie stosowane do tego samego zestawu danych [41]. Algorytm AdaBoost wybiera tylko te cechy, o których wiadomo, że poprawiają moc predykcyjną modelu, zmniejszając wymiarowość i potencjalnie poprawiając czas wykonania, ponieważ nieistotne cechy nie muszą być obliczane. Na początku algorytm nadaje wszystkim próbkom równe wagi i prawdopodobieństwa wylosowania w kolejnej iteracji. Po każdej iteracji prawdopodobieństwa są aktualizowane. Próbką, która została poprawnie sklasyfikowana, ma mniejsze prawdopodobieństwo, że zostanie wylosowana w następnej iteracji, a błędnie sklasyfikowana próbka ma większe prawdopodobieństwo. W rezultacie klasyfikator w dalszej części serii tworzy zestaw treningowy składający się z trudnych do sklasyfikowania próbek. Metody uczenia jednoklasowego (OOC) - czyli mającego na celu identyfikację obiektu jednej, określonej klasy, poprzez uczenie się przede wszystkim ze zbioru zawierającego tylko obiekty tej klasy, mają na celu zwalczanie problemu overfittingu. Występuje on przypadku większości klasyfikatorów uczących się na niezrównoważonych danych. Osiągane jest to poprzez podejście do tego problemu z punktu widzenia uczenia nienadzorowanego [24-26]. Algorytmy jednoklasowe są konstruowane w taki sposób, aby rozpoznawać próbki z danej klasy i odrzucać próbki z innych klas.

2.6.3 Podejścia hybrydowe

Metody hybrydowe mają na celu rozwiązanie znanych problemów spowodowanych metodami próbkowania danych, metodami wyboru cech, metodami wrażliwymi na koszty i podstawowymi algorytmami uczenia się (takimi jak Naive Bayes [42]). W niektórych przypadkach podgrupy metod na poziomie danych lub podgrupy metod na poziomie algorytmu można łączyć jako ogólną metodę rozwiązywania problemu niebalansowania klas. Na przykład popularny klasyfikator losowego lasu (Random Forest) jest wersją oryginalnego algorytmu losowego lasu decyzyjnego (Random Decision Forest) i jest zintegrowanym algorytmem uczenia się, który dodatkowo implementuje Bagging [43].

2.7 Metody oparte o selekcję cech

Pojęcie przekleństwo wymiarowości mówi, że jeśli wiele cech jest zaszumionych, koszt użycia klasyfikatora może być bardzo wysoki, a wydajność może być poważnie zaniżona [9]. Ponieważ

problemowi z nierównowagą klas często towarzyszy problem dużej wymiarowości zbioru danych, zastosowanie technik selekcji cech jest koniecznym działaniem [9]. Pomysłowe techniki próbkowania i metody algorytmiczne mogą nie wystarczyć do walki z wysokowymiarowymi problemami nierównowagi klas. Van der Putten i van Someren przeanalizowali zbiory danych z CoIL Challenge 2000 i stwierdzili, że wybór cech był bardziej istotny dla dobrych wyników niż wybór algorytmu klasyfikacji i najbardziej pomogły w walce z problemem nadmiernego dopasowania [26]. Forman odnotował podobną obserwację dotyczącą wysoce nie zrównoważonych problemów klasyfikacji tekstu i stwierdził, że „żaden stopień sprytnej indukcji nie może zrekompensować braku sygnału predykcyjnego w przestrzeni wejściowej” [27-29]. Badania te pokazują, że w wielowymiarowych zbiorach danych, sam dobór cech może zwalczyć problem nierównowagi klas.

W ostatnich latach, radzenie sobie z nie zrównoważonymi zbiorami danych za pomocą selekcji cech stało się popularne wśród społeczności zajmujących się eksploracją danych i uczeniem maszynowym [13]. Wspomniane wcześniej techniki koncentrują się na próbkowaniu danych uczących w celu przezwyciężenia nie zrównoważonego rozkładu klas. Metoda redukcji cech, taka jak selekcja cech, przyjmuje inne podejście do przezwyciężenia problemu. Ogólna koncepcja polega na uzyskaniu podzbioru cech, które optymalnie korygują dysproporcje między klasami w zbiorze danych i wybierają najlepsze cechy, które reprezentują obie klasy.

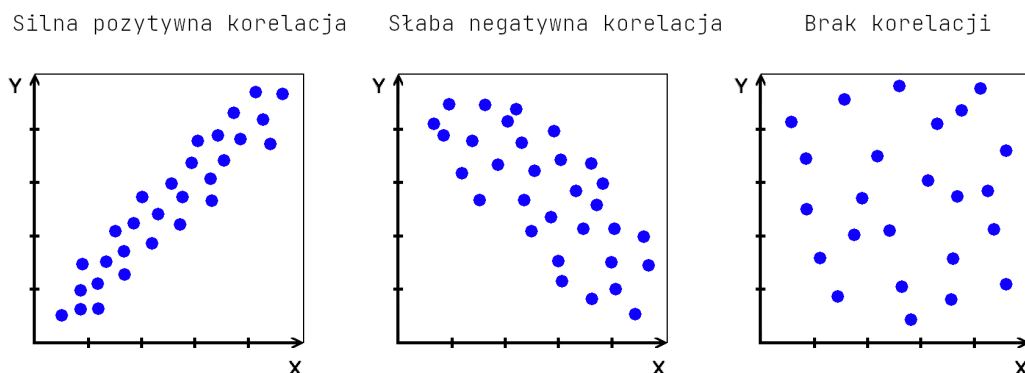
Przedstawione w rozdziałach 2.7.1 - 2.7.5 algorytmy należą do tradycyjnych, szeroko używanych metod selekcji cech [8]. Omówione strategie należą do tak zwanych filtrów, czyli metod rankingowych. Jest to najbardziej naturalne podejście do rozpatrywanego tematu, gdyż opisane algorytmy nie są zależne od wbudowanego klasyfikatora. Pozwoli to również na ich kompleksowe i obiektywne porównanie.

2.7.1 Correlation coefficient

Korelacja to miara liniowej zależności pomiędzy dwoma zmiennymi losowymi. Jest to więc po prostu miara tego, jak silnie jedna zmienna zależy od drugiej. Jest to zazwyczaj bardzo użyteczna właściwość - w przypadku dwóch, silnie skorelowanych zmiennych, posiadając informacje o jednej zmiennej można przewidzieć wartości innej zmiennej. W przypadku liniowych modeli uczenia maszynowego, częstym celem będzie znalezienie elementów silnie skorelowanych ze zmienną losową opisującą przynależność do klasy. Jednakże, dwie silnie skorelowane ze sobą zmienne dostarczają też redundantnych informacji. Zasadniczo można dokonać poprawnego sklasyfikowania z pomocą tylko jednej z tych atrybutów. Usunięcie drugiej może więc pomóc w zmniejszeniu wymiarowości i zbędnego szumu [32].

Współczynnik korelacji Pearsona to algorytm, który określa poziom zbieżności liniowej pomiędzy zmiennymi. Wynikiem tej metody są wartości od -1 do 1. Bezwzględna wartość współczynnika określa siłę zależności liniowej - wartości bliższe 1 wskazują na silniejszy związek. Znak współczynnika wskazuje kierunek zależności: znak dodatni wskazuje, że dwie zmienne rosną lub maleją względem siebie (pod względem korelacji), a znak ujemny wskazuje, że jedna zmienna

rośnie, a druga maleje [8]. Interpretacje wyników korelacji na przykładowych zbiorach danych zostały przedstawione na grafice 2.3.



Rysunek 2.3: Interpretacje współczynnika korelacji dla różnych zbiorów danych

Ogólna postać testu przedstawiona została w równaniu 2.1:

$$r = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2} \sqrt{\sum_{i=1}^n (y_i - \bar{y})^2}}, \quad (2.1)$$

gdzie:

- n jest liczebnością próbki
- x_i oraz y_i są indywidualnymi próbkami indeksowanymi po i
- $\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$ - jest średnią arytmetyczną (podobnie dla \bar{y})

Współczynnik korelacji Pearsona można wykorzystać do oceny związku między więcej niż dwiema zmiennymi, obliczając macierz relacji między każdą parą zmiennych w zbiorze danych. Rezultatem jest symetryczna macierz zwana macierzą korelacji.

2.7.2 Chi-square

Chi-kwadrat jest testem statystycznym mierzącym niezależność cechy od etykiety klasy. Test chi-kwadrat mierzy zależność między zmiennymi stochastycznymi, więc użycie tej metody "usuwa": cechy, które z największym prawdopodobieństwem są niezależne od klasy, a zatem nie mają znaczenia dla klasyfikacji. Metoda polega na obliczeniu metryki χ^2 pomiędzy wartością docelową a cechą i wyborze zmiennej o maksymalnym wyniku testu [30, 31].

Ogólna postać testu została przedstawiona w równaniu 2.2:

$$\chi^2 = \sum_{i=1}^n \left(\frac{O_i - E_i}{E_i} \right)^2, \quad (2.2)$$

gdzie:

- O_i jest wartością mierzoną
- E_i jest wartością oczekiwaną
- n jest liczbą pomiarów.

Forman zauważył, że ten test może zachowywać się nieprawidłowo, gdy spodziewana jest niewielka liczba cech; jest to dość powszechne w przypadku niezrównoważonych zbiorów danych [28]. Chociaż test chi-kwadrat dobrze uogólnia dane dyskretne, nie radzi sobie dobrze się podczas testowania danych ciągłych [12].

2.7.3 Information Gain

Entropia warunkowa to entropia po podziale zbioru przy pomocy danego atrybutu [12]. Dla danego atrybutu a , entropia warunkowa wyraża się wzorem 2.3:

$$Ent(S|a) = \sum_{j=1}^p \frac{n_{s_j}}{n} Ent(S_j), \quad (2.3)$$

gdzie:

- p to liczba wartości atrybutu a
- S_j to zbiór przykładów z wartością atrybutu v_j
- n_{s_j} to liczebność zbioru S_j
- $Ent(S_j)$ to entropia zbioru S_j , wyrażona wzorem 2.4:

$$Ent(S) = - \sum_{i=1}^k p_i \log_2 p_i, \quad (2.4)$$

gdzie:

- p_i to prawdopodobieństwo przynależności do klasy i – tej
- S to zbiór przykładów
- k liczba klas

Innymi słowy, im mniejsza wartość entropii warunkowej, tym większa jednorodność podziału [32]. Information Gain mierzy różnicę między entropią etykiet klas a entropią warunkową etykiet klas dla danej cechy [12]. Metoda ta ocenia przyrost informacji przy użyciu atrybutu. Dla danego atrybutu a :

$$IG(S, a) = Ent(S) - Ent(S|a) \quad (2.5)$$

Podobnie jak test chi-kwadrat, uogólnia atrybuty dyskretne, ale nie radzi sobie atrybutami z danymi ciągłymi. Ponadto preferuje atrybuty o dużej liczbie wartości i może prowadzić do przeuczenia [32]. Problemy te rozwiązuje zmodyfikowana wersja algorytmu - Gain Ratio.

2.7.4 Relief i ReliefF

Algorytmy Relief są jednym z najskuteczniejszych, opracowanych dotychczas metod filtrujących [12]. Większość metod filtrowania opracowanych w eksploracji danych i uczeniu maszynowym zakłada warunkową niezależność atrybutów. Algorytmy Relief są bardzo przydatne, ponieważ nie zakładają, że atrybuty są od siebie niezależne. Te algorytmy są zależne od kontekstu. Kiedy istnieje silny związek między atrybutami, jakość atrybutów może być poprawnie oszacowana, co sprawia, że algorytm ten jest jednym z najbardziej efektywnych algorytmów przetwarzania wstępnego [12].

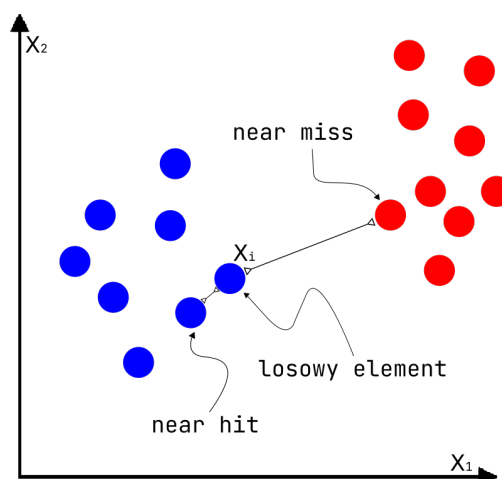
Podstawową ideą algorytmów Relief jest oszacowanie jakości cech na podstawie tego, jak dobrze cecha może rozróżniać instancje, które są blisko siebie. Relief oblicza punktację dla każdej cechy, którą można następnie zastosować do selekcji cech. Punktacja opiera się na wyliczeniu różnic wartości cechy między parami najbliższych sąsiadów. Jeśli różnica wartości cechy zostanie zaobserwowana w sąsiedniej parze instancji z tą samą klasą, wynik funkcji maleje. Jeżeli natomiast różnica wartości cechy zostanie zaobserwowana w sąsiedniej parze instancji z różnymi wartościami klas, wynik funkcji rośnie [47]. Algorytm został pierwotnie zaprojektowany do zastosowania w problemach klasyfikacji binarnej. Relief zainspirował rodzinę algorytmów wyboru cech - RBA (Relief Based Algorithms), w tym algorytm ReliefF [47]. Są one przystosowane do działania w problemach wieloklasowych lub zbiorach z niekompletnymi danymi.

Metoda na wejściu przyjmuje wektor wartości cechy i klasy. Algorytm będzie powtarzany m razy i rozpoczyna się z tablicą wag W o długości równej ilości cech, wypełnioną zerami. Podczas każdej iteracji, algorytm rozpatruje wektor cech X należący do losowej instancji i wektory cech instancji najbliższe X (według odległości euklidesowej) z każdej cechy. Najbliższa instancja tej samej klasy nazywana jest *prawie trafioną* (*near hit*), natomiast najbliższa instancja innej klasy - *prawie spudłowaną* (*near miss*). Wektor wag aktualizowany jest według wzoru 2.6:

$$W_i = W_i - (x_i - nearHit_i)^2 + (x_i - nearMiss_i)^2, \quad (2.6)$$

gdzie:

- W_i to i – ty element wektora wag W
- x_i to i – ty element wektora X
- $nearHit_i$ to i – ty element wektora *near hit*
- $nearMiss_i$ to i – ty element wektora *near miss*



Rysunek 2.4: Wybór elementów *near hit* oraz *near miss* w każdej iteracji działania algorytmu

Waga danej cechy maleje, jeżeli różni się ona od tej cechy w pobliskich instancjach tej samej klasy bardziej, niż pobliskie instancje innych klas, a wzrasta w przeciwnym przypadku. Po m iteracjach, każdy element wektora W jest dzielony przez m . Tworzy on wtedy ranking cech [47].

Zaletą metod RBA jest to, że nie są zależne od heurystyki, działają w czasie wielomianowym niskiego rzędu, są odporne na zakłócenia, a także nadają się do danych binarnych lub ciągłych [48, 49].

2.7.5 ANOVA

ANOVA - od angielskiego analysis of variance, czyli analiza wariancji. Jest to metoda, która wyjaśnia, z jakim prawdopodobieństwem wyodrębnione czynniki mogą być powodem różnic między obserwowanymi średniami grupowymi. Algorytm polega na porównaniu wariancji międzygrupowej do wariancji wewnątrzgrupowej. ANOVA to jeden z algorytmów statystyki F (F-test) - od nazwiska twórcy, Ronalda Fishera [51].

Analizę wariancji można podzielić na trzy grupy analiz:

- **jednoczynnikowa analiza wariancji** - wpływ jednego czynnika międzygrupowego na zmienną zależną
- **wieloczynnikowa analiza wariancji** - wpływ kilku czynników międzygrupowych na zmienną zależną
- **analiza wariancji dla czynników wewnątrzgrupowych** - wpływ czynnika wewnątrzgrupowego na zmienną zależną, tzw. „powtarzane pomiary”

Analiza wariancji to stosunek wariancji, obliczona pomiędzy badanymi grupami a średnią wariancją, zaobserwowaną wewnątrz grup. Analiza ta jest metodą statystyczną pozwalającą na podział zaobserwowanej wariancji wyników na oddzielne części. Analizowana jest wariancji przypadająca na każdy z analizowanych czynników jak również wariancji błędu. Idealna sytuacja

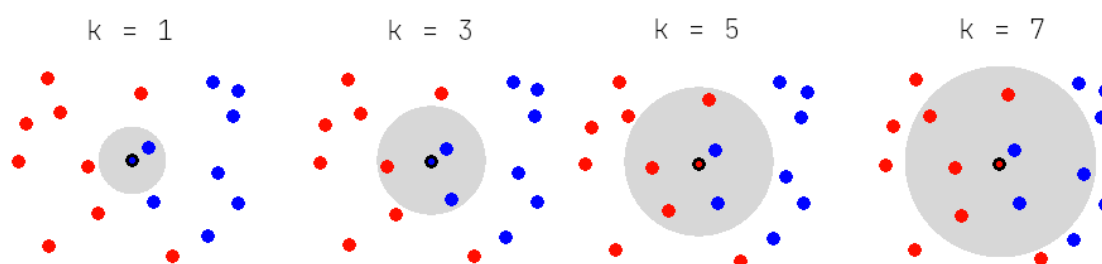
ma miejsce wtedy, gdy wariancja międzygrupowa jest duża natomiast wariancja wewnątrzgrupowa - mała [50, 51].

3 Założenia i plan eksperymentu

Obiektywne porównanie przytoczonych algorytmów cech wymaga przeprowadzenia szeregu eksperymentów porównujących skuteczność popularnych metod selekcji cech. Praca swoim zakresem obejmie eksperymenty przeprowadzone na kilku, wybranych zbiorach danych. Zakłada się użycie zbiorów rzeczywistych - to znaczy zebranych w ramach rzeczywistych pomiarów. Bazy danych, użyte w ramach badań implikują skupienie na problemach dwuklasowych. Hipoteza, z którą twórca będzie konfrontować wyniki eksperymentów, to założeniem że wszystkie, badane metody selekcji poradzą sobie podobnie z postawionym zadaniem, a poza względami wydajnościowymi, nie ma znaczenia funkcja, która zostanie użyta. Technologia, w jakiej zostaną przeprowadzone doświadczenie to język Python w wersji 3.8 oraz biblioteki `scikit-learn` (<https://scikit-learn.org>), `numpy` (<https://numpy.org>). i `pandas` (<https://pandas.pydata.org/>)

3.1 Generowanie wyników

Wybrane przez autora metody selekcji cech należą do grupy tak zwanych filtrów. Tworzą one ranking cech, przydatny do zdefiniowania cech, które będą używane przez algorytm w celu przeprowadzenia klasyfikacji. Do prawidłowego porównania badanych algorytmów należy sprawdzić ich wyniki w połączeniu z całym procesem klasyfikowania. Zdecydowano się na klasyfikator KNN - K Najbliższych Sąsiadów. W metodzie tej, klasyfikowany obiekt przydzielany jest do klasy, do której należy większość z jego sąsiadów [51, 52].



Rysunek 3.1: Graficzna reprezentacja działania algorytmu KNN

Algorytm przyjmuje zbiór danych uczących zawierający elementy, z których każda ma przypisaną klasę oraz wektor cech. Dany jest element C z przypisanym wektorem cech $X_1 \dots X_n$ dla którego prognoza odbywa się w następujący sposób:

1. Porównanie wartości cech dla elementu C z wartościami cech dla każdego elementu w zbiorze uczącym.
2. Wybór k (ustalona z góry liczba) najbliższych do C elementów zbioru uczącego.

3. Uśrednienie wartości klasy dla wybranych elementów, w wyniku czego uzyskiwana jest prognoza.

3.2 Ocena działania algorytmów

Określenie jakości działania algorytmu może stanowić badany przypadek problemem. Dokładność (accuracy) używana jako metryka ewaluacji może być w takim przypadku niewystarczająca, gdyż nawet model o skuteczności 95% - co jest na ogół wartością bardzo dobrą - mógłby nie rozpoznawać żadnego elementu klasy mniejszościowej (w przypadku rozkładu 5 - 95). Metrykami, które dostarczą bardziej wartościowe dane są:

- Macierz konfuzji: tabela pokazująca prawidłowe prognozy i typy nieprawidłowych przewidywań.
- Precyzja: liczba prawdziwie pozytywnych wyników podzielona przez wszystkie pozytywne przewidywania. Precyzja jest również nazywana pozytywną wartością predykcyjną. Jest miarą dokładności klasyfikatora. Niska precyzja wskazuje na dużą liczbę fałszywych wyników.
- Czułość: liczba prawdziwie pozytywnych wyników podzielona przez liczbę dodatnich wartości w danych testowych. Jest miarą kompletności klasyfikatora. Niska czułość wskazuje na dużą liczbę fałszywie negatywnych wyników.
- F1 Score: średnia harmoniczna precyzji i czułości, wyrażona wzorem 3.1:

$$F1Score = \frac{2 * (Recall * Precision)}{Recall + Precision}. \quad (3.1)$$

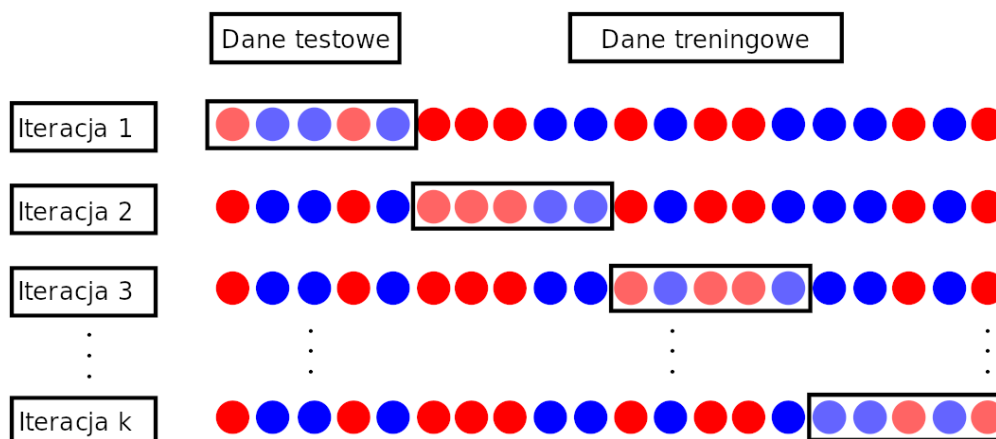
Pod uwagę brane będą funkcja straty (loss), która informuje o dopasowaniu modelu do danych, oraz dokładności (accuracy), która określa skuteczność klasyfikacji. W celu określenia, która z testowanych metod daje najlepsze wyniki klasyfikacji wykorzystany zostanie test statystyczny - test Wilcoxa [44]. Do jego wykonania użyte zostaną wartości dokładności uzyskane dla każdej z badanych metod.

3.3 Generowanie zbioru treningowego i testowego

W początkowym etapie uczenia maszynowego, programista dysponuje jedynie spójnym zbiorem danych. Jednym z kluczowych kroków w procesie jest podział zbioru na zbiory: treningowy i testowy. Jest to konieczne w celu wydzielania fragmentów, na których klasyfikator będzie się uczył i tych, na których nastąpi testowanie wyuczonego już klasyfikatora. Pominięcie tego kroku może skutkować błędnymi, wysokimi wynikami dokładności (accuracy). Istnieje kilka metod podziału zbioru. Najprostrzym możliwym podejściem jest losowy podział zbioru na dwie części z zachowaniem proporcji ilościowej. Pozwala to na uzyskanie dwóch podzbiorów, lecz istnieje przy tym ryzyko, że losowość dokonała się w sposób, który wykaże błędną, wyższą jakość klasyfikacji. Rozwiązaniem tego problemu jest wielokrotny podział. Metodą, która zapewnia wielokrotny,

sprawiedliwy podział, eliminujący prawdopodobieństwo występowania tych samych próbek w różnych zbiorach uczących jest *K-Fold Cross-Validation* [45].

W podejściu krzyżowym, zbiór dzielony jest losowo na k równych podzbiorów. Następnie kolejno każdy z nich używa się jako zbiór testowy, a połączoną resztę - jako zbiór uczący. Finalnie, rezultaty uśrednia się w celu uzyskania jednorodnego wyniku.



Rysunek 3.2: Graficzna reprezentacja działania algorytmu walidacji krzyżowej

W ramach przeprowadzonych eksperymentów posłużono się funkcją *KFold* z biblioteki *scikit-learn* [46]. Umożliwia ona użycie ziarna losowości, co zapewnia możliwość powtórzenia uzyskanych w ten sposób wyników.

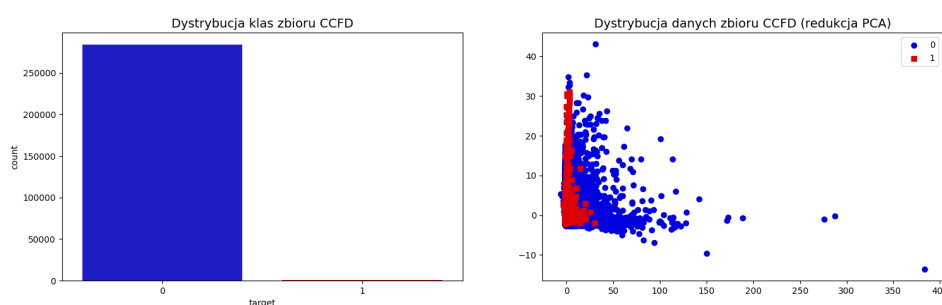
3.4 Zbiory danych

W ramach przeprowadzonych doświadczeń posłużono się ponad stoma zbiorami danych. Wszystkie charakteryzowały się wysokim niezbalansowaniem i wykazywały nadreprezentację jednej bądź wielu cech. Rozdziały 3.4.1 - 3.4.4 zawierają szczegółowy opis dużych, starannie przygotowanych w ramach pracy zbiorów z różnych dziedzin nauki. Rozdział 3.4.5 skrótowo opisuje pozostałe zbiory danych.

3.4.1 Credit Card Fraud Detection

Zbiór danych zawiera informacje o transakcjach dokonanych kartami kredytowymi we wrześniu 2013 roku, przez europejskich posiadaczy kart. Dataset składa się z transakcji, które miały miejsce w ciągu dwóch dni, w których miały miejsce 492 oszustwa z 284 807 wszystkich transakcji. Zbiór jest wysoce niezbalansowany, a klasa pozytywna (oszustwa) stanowi 0,172% wszystkich transakcji [53].

Elementy zbioru składają się tylko z danych liczbowych, które są wynikiem transformacji PCA. Jest to podyktowane względami na kwestie poufności - twórcy nie są w stanie dostarczyć oryginalnych wartości ani dodatkowych informacji o danych. Cechy V_1, V_2, \dots, V_{28} są głównymi składnikami uzyskanymi za pomocą PCA, jedynymi cechami, które nie zostały przekształcone za pomocą PCA, są „Time” i „Amount”. Wartość „Time” zawiera sekundy, które upłynęły między każdą transakcją a pierwszą transakcją w zbiorze danych. Funkcja „Amount” to kwota transakcji. Cecha „Class” jest zmienną odpowiedzi i przyjmuje wartość 1 w przypadku oszustwa i 0 w innym przypadku. Dystrybucja klas oraz danych została ukazana na rysunku 3.3. W celu ukazania dystrybucji danych na dwuwymiarowym wykresie, zastosowano ekstrakcję cech metodą 2-PCA.



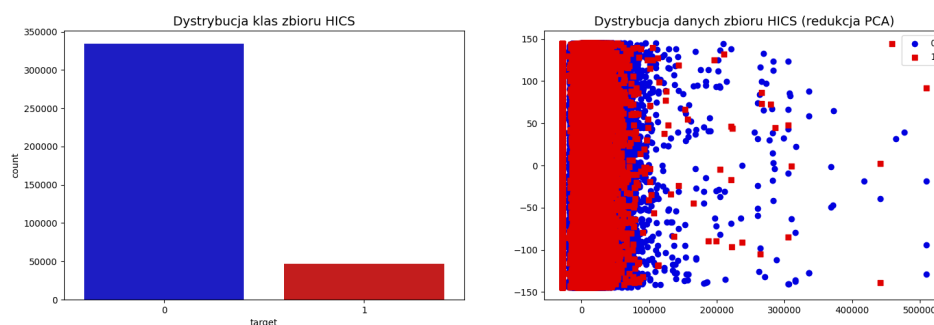
Rysunek 3.3: Dystrybucja klas i danych dla zbioru Credit Card Fraud Detection

Zbiór został pozyskany za pośrednictwem platformy Kaggle (<https://www.kaggle.com/>).

3.4.2 Health Insurance Cross Sell

Zbiór został pierwotnie stworzony w ramach konkursu. Celem wyzwania było przewidywanie prawdopodobieństwa, że klient firmy ubezpieczeniowej będą zainteresowani ubezpieczeniem komunikacyjnym oferowanym przez firmę [62]. Każdy wiersz odpowiada określonej posiadaczowi polisy, a kolumny opisują ich cechy. Zmienna docelowa jest tu dogodnie nazywana wynikiem (*result*) i wskazuje, czy ubezpieczający będzie zainteresowany zakupem polisy.

Zbiór danych został dobrze udokumentowany przez twórców. Każdy klient został opisany dwunastoma cechami i są to: płeć (gender), wiek (age), posiadanie prawa jazdy (driving_license), region zamieszkania klienta (region_code), informacja o poprzednim ubezpieczeniu (previously_insured), wiek pojazdu (vehicle_age), informacja o uszkodzeniach pojazdu (vehicle_damage), koszty ubezpieczenia (annual_premium), kanał kontaktu z klientem (policy_sales_channel) i czas jaki klient jest związany z firmą (vintage). Dane o płci, wieku samochodu i uszkodzeniach pojazdu są danymi tekstowymi. Dystrybucja klas oraz danych została ukazana na rysunku 3.4. W celu ukazania dystrybucji danych na dwuwymiarowym wykresie, zastosowano ekstrakcję cech metodą 2-PCA.



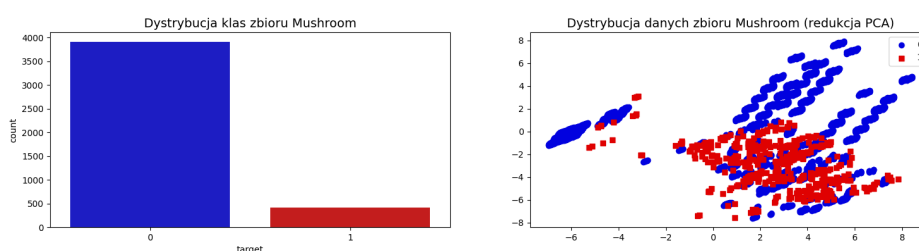
Rysunek 3.4: Dystrybucja klas i danych dla zbioru Health Insurance Cross Sell

Zbiór został pozyskany za pośrednictwem platformy Kaggle (<https://www.kaggle.com/>).

3.4.3 Mushroom Classification

Zestaw danych zawiera opisy próbek różnych gatunków grzybów, zaczerpnięte z The Audubon Society Field Guide to North American Mushrooms (1981) [62]. Każdy gatunek jest określony jako zdecydowanie jadalny, zdecydowanie trujący lub o nieznanym jadalności (niezalecany). Ta ostatnia klasa została połączona z klasą trującą. Przewodnik jasno stwierdza, że nie ma prostej zasady określania jadalności grzyba - eksperyment pozwoli stwierdzić czy jest tak w istocie. Każdy gatunek grzyba opisany jest za pomocą 23 cech, wśród których znajdują się cechy takie jak rozmiar i kształt kapelusza, rozmiar blaszek i przerw pomiędzy nimi, kolor, wysokość, typ pierścienia wokół pnia itp. Wszystkie cechy opisane są zmiennymi tekstowymi. Sam zbiór nie wykazuje cech niezbalansowania. Został odpowiednio dostosowany do celów eksperymentu, zostało to opisane w rozdziale 3.5.3.

Dystrybucja klas oraz danych została ukazana na rysunku 3.5. W celu ukazania dystrybucji danych na dwuwymiarowym wykresie, zastosowano ekstrakcję cech metodą 2-PCA.

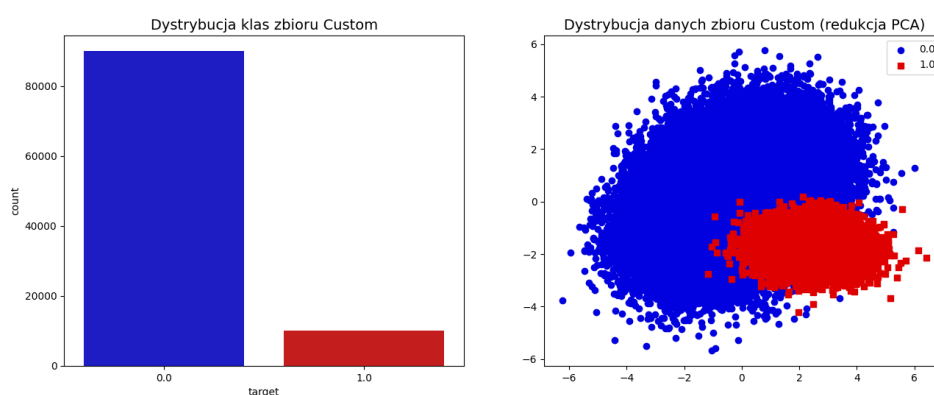


Rysunek 3.5: Dystrybucja klas i danych dla zbioru Mushroom

Zbiór został pozyskany za pośrednictwem platformy Kaggle (<https://www.kaggle.com/>).

3.4.4 Zbiór „Custom”

Zbiór custom został wygenerowany sztucznie z pomocą funkcji *make_classification* biblioteki *scikit-learn* [61]. Pozwoliło to na dobranie parametrów w taki sposób, aby zbiór odpowiadał jak najlepiej założeniom projektowym. Dataset składa się 100000 elementów. Kolumna *target* wskazuje na przypasowanie elementu do jednej z dwóch klas. Zbiór jest niezbalansowany, jedynie 10% elementów należy do klasy pozytywnej. Każdy element opisany jest za pomocą 20 cech, z czego trzy z nich są najbardziej informatywne, a jedna jest zupełnie zbędna. Zbiór nie zawiera pustych danych. Dystrybucja klas oraz danych została ukazana na rysunku 3.6. W celu ukazania dystrybucji danych na dwuwymiarowym wykresie, zastosowano ekstrakcję cech metodą 2-PCA.



Rysunek 3.6: Dystrybucja klas i danych dla zbioru Custom

3.4.5 Inne zbiory danych

W celu poprawnego i obiektywnego porównania wszystkich metod selekcji wykonano eksperymenty na dodatkowych 111 zbiorach udostępnionych przez platformę KEEL [63]. W tym celu stworzono program automatycznie importujący i przygotowujący zbiór do wykorzystania przez testowane algorytmy. Każdy ze zbiorów posiada pewne cechy wspólne - wszystkie one zostały stworzone w podobnym formacie, zawierają cechy opisane numerycznie i dwie klasy - opisane za pomocą etykiet „positive” oraz „negative”. Ponadto klasą mniejszościową jest zawsze klasa „positive”.

3.5 Przygotowanie danych

Dane wysokiej jakości są wymagane wstępnie dla modeli predykcyjnych. Wstępne przetwarzanie i czyszczenie danych to ważne zadania, które należy wykonać, zanim zestaw danych będzie mógł zostać użyty do uczenia modelu. Nieprzetworzone dane są zwykle zaszumione i mogą zawierać błędne wartości lub luki. Modelowanie przy użyciu tego typu danych może dawać mylące wyniki.

Szczególnie na takie zagrożenia narażone są dane rzeczywiste i zbiory zbierane z różnych źródeł. Typowe problemy z jakością danych to:

- Niekompletność: dane nie mają atrybutów lub zawierają brakujące wartości.
- Zakończenia: dane zawierają błędne rekordy lub elementy odstające.
- niespójność: dane zawierają rekordy powodujące niezgodności.

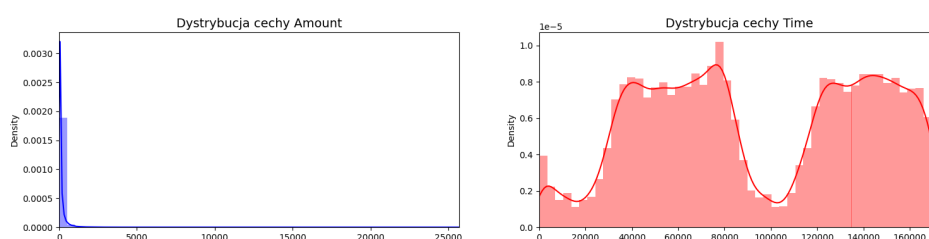
W celu odpowiedniego, wstępnego przetworzenia danych, do każdego zbioru należy podejść indywidualnie. Są jednak metody, które pozwalają poradzić sobie z większością napotkanych problemów. Do popularnych metod radzenia sobie z problemami wynikającymi ze źle przygotowanych bazami danych należą:

- Uzupelnienie brakujących wartości.
- Likwidowanie wartości odstających (ang. outliers).
- Standaryzacja poprzez normalizację lub dyskretyzację.
- Redukcja wymiarów poprzez selekcję i ekstrakcję cech.
- Równoważenie danych poprzez usuwanie przypadków klas większościowych lub nadpróbkowanie klas mniejszościowych.
- Transformacja zmiennych (np. liniowa).
- Dodawanie nowych zmiennych w celu zwiększenia liczby cech (np. iloczynów istniejących zmiennych).
- Podział danych na dane treningowe i dane testowe.

Zbiory użyte w ramach eksperymentów należało w większości poddać wstępnej obróbce. Rozdziały 3.5.1 - 3.5.5 opisują działania podjęte w celu poprawienia jakości danych.

3.5.1 Preprocessing zbioru CCFD

W zbiorze *Credit Card Fraud Detection* nie występują puste wartości. Większość cech została wcześniej poddana transformacji metodą PCA czego efektem ubocznym jest ich wyskalowanie, które konieczne jest w przypadku użycia tej metody. Cechami wyróżniającymi się są „Time” oraz „Amount”. Dystrybucja wartości dla tych cech ukazana jest na wykresach 3.7.



Rysunek 3.7: Dystrybucja wartości dla cech *Time* oraz *Amount*

Wartości te należało przeskalować, aby nie odstawały od innych danych. Użyto w tym celu metody *RobustScaler* z biblioteki *scikit-learn* [54].

3.5.2 Preprocessing zbioru HICS

W zbiorze *Health Insurance Cross Sell* nie występują puste wartości. Większość cech to cechy binarne, które nie potrzebują opróbki. Jak wspomniano w rozdziale 3.4.2, dane o płci, uszkodzeniach pojazdu i wieku samochodu są danymi tekstowymi. Kolumna *gender* przyjmuje wartości *Male* oraz *Female*, natomiast kolumna *vehicle_damage* - *yes* oraz *no*. Wartości te zostały zmienne na dane binarne: „0” i „1”. Kolumna opisująca wiek pojazdu posiada 3 możliwe wartości tekstowe, które zostały zmapowane na wartości liczbowe: „0”, „1” i „2”.

3.5.3 Preprocessing zbioru Mushroom

Zbiór *Mushroom* był początkowo zbiorem zbalanowanym. W celu utworzenia zbioru niezbalansowanego, usunięto z niego 90% elementów zakwalifikowanych jako grzby trujące (zostawiając co dziesiąty element oznaczony klasą „p”). Ponadto, wszystkie cechy zbioru opisane zostały etykietami słownymi. Stworzyło to konieczność przepisania wszystkich elementów zbioru na etykiety numeryczne. Posłużono się w tym celu funkcją *LabelEncoder* zbioru *scikit-learn* [64]. Metoda ta enkoduje każdy element listy wartością pomiędzy 0 a $n - 1$ gdzie n to liczba wariantów cechy.

3.5.4 Preprocessing zbioru Custom

Ponieważ zbiór *Custom* został wygenerowany sztucznie, wszystkie parametry zostały dobrane tak, by nie zachodziła potrzeba jego ponownej obróbki.

3.5.5 Preprocessing pozostałych zbiorów

W zbiorze *Health Insurance Cross Sell* nie występują puste wartości. Większość cech to cechy binarne, które nie potrzebują opróbki. Jak wspomniano w rozdziale 3.4.2, dane o płci, uszkodzeniach pojazdu i wieku samochodu są danymi tekstowymi. Kolumna *gender* przyjmuje wartości *Male* oraz *Female*, natomiast kolumna *vehicle_damage* - *yes* oraz *no*. Wartości te zostały zmienne na dane binarne: „0” i „1”. Kolumna opisująca wiek pojazdu posiada 3 możliwe wartości tekstowe, które zostały zmapowane na wartości liczbowe: „0”, „1” i „2”.

3.6 Eksperymenty

Metody opisane w ramach rozdziałów 2.7.1 - 2.7.5 zostały zaimplementowane przy pomocy bibliotek *scikit-learn* oraz *pandas*. Algorytmy ANOVA, χ^2 oraz *Information Gain* mają swoją gotową implementację w bibliotece *scikit-learn* - kolejno funkcje *f_classif* [55], *chi2* [56] i *mutual_info_classif* [57]. Zostały one wykorzystane w ramach metody *SelectKBest*, umożliwiającej zdefiniowanie metody selekcji cech oraz docelowej ilości cech. Eksperymenty zostały powtórzone kilka razy z różnymi ustawieniami w celu ustalenia optymalnej liczby cech dla każdego

zbioru. Algorytm *Relieff* zaimplementowany został przy użyciu biblioteki *sklearn_relief* [58]. Metoda *Correlation Coefficient* zaimplementowana została przy użyciu funkcji *corr* z pakietu *pandas*, zwracającej korelację pomiędzy parami wszystkich kolumn danego zbioru.

W ramach wszystkich zbiorów danych przeprowadzony został eksperyment badający skuteczność klasyfikacji bez wykonywania wcześniejszej selekcji cech oraz biorący pod uwagę różną ilość cech w procesie klasyfikacji.

W celu ułatwienia sprawnego i obiektywnego porównania wszystkich metod, w ramach pracy stworzono bibliotekę zawierającą funkcje przygotowujące dane, dzielące zbiór na podzbiory, przeprowadzające klasyfikacje oraz wyliczające wyniki eksperymentów. Rezultatem każdego doświadczenia jest plik zatytułowany metodą selekcji cech użytą do klasyfikacji. Plik ten zawiera nazwę zbioru użytego do klasyfikacji, sumaryczną ilość cech oraz ilość cech po przeprowadzonej selekcji, oraz dokładność, precyzję, macierz konfuzji, wartość FPT i wartość TPR w czterech wariantach - przed selekcją i po selekcji cech oraz z wykorzystaniem metody *walidacji krzyżowej* oraz z wykorzystaniem autorskiej metody dzielenia zbioru opisanej w rozdziale 3.3.

Łącznie przeprowadzono 66 eksperymentów, po 12 dla każdej opisanej metody selekcji cech. Eksperymenty zawierają każdą permutację z zadanych zagadnień:

- Porównanie wyników dla każdego z 3 zbiorów danych.
- Porównanie wyników dla danych poddanych selekcji cech i klasyfikacji dla całego zbioru.
- Porównanie wyników dla każdej z 5 metod selekcji.
- Porównanie wyników dla każdej z 2 metod podziału zbioru.

Wyniki eksperymentów opisane zostały w rozdziale 4.

4 Wyniki

5 Wnioski

6 Bibliografia

- [1] Feature Selection for High-Dimensional and Imbalanced Biomedical Data Based on Robust Correlation Based Redundancy and Binary Grasshopper Optimization Algorithm, Garba Abdulrauf Sharifai and Zurinahni Zaino; 2020
- [2] A Survey on Evolutionary Computation Approaches to Feature Selection, Bing Xue; Mengjie Zhang; Will N. Browne; Xin Yao, 2015
- [3] Correlation-based Feature Selection for Machine Learning, Mark A Hall, University of Waikato, 1999
- [4] Efektywne metody selekcji cech i rozwiązywania problemu wieloklasowego w nadzorowanej klasyfikacji danych, Wiesław Chmielnicki, IPPT PAN, 2012
- [6] Zastosowanie wybranych metod przekształcania i selekcji danych oraz konstrukcji cech w zadaniach klasyfikacji i klasteryzacji, Piotr Płoński, Politechnika Warszawska, 2016.
- [7] Analiza i rozwój metod selekcji cech dla dużych problemów klasyfikacyjnych, Jakub Piątkowski, Uniwersytet Mikołaja Kopernika, 2006
- [8] Feature Selection in Imbalance data sets, International Journal of Computer Science Issues, 2013
- [9] G. Weiss and F. Provost, „Learning when Training Data Are Costly: The Effect of Class Distribution on Tree Induction,” J. Artificial Intelligence Research, vol. 19, 2003
- [10] Paweł Ziemia, Redukcja wymiarowości i selekcja cech w zadaniach klasyfikacji i regresji z wykorzystaniem uczenia maszynowego, 2012
- [11] Wasikowski, M.; Chen, X.-w. Combating the small sample class imbalance problem using feature selection. IEEE Trans. Knowl. Data Eng. 2009
- [12] Deepika Tiwari, Handling Class Imbalance Problem Using Feature Selection. International Journal of Advanced Research in Computer Science & Technology, June 2014
- [13] C. Elkan, „Magical Thinking in Data Mining: Lessons from CoIL Challenge 2000” Proc. ACM SIGKDD '01, 2001
- [14] Isabelle Guyon & Andre Elisseeff, An Introduction to Variable and Feature Selection, Journal of Machine Learning Research, 2003
- [15] Moayedikia; Yeoh W.G.; Jensen R; Feature selection for high dimensional imbalanced class data using harmony search. Eng. Appl. Artif. Intell. 2017
- [16] Miron Bartosz Kurs, Stabilne i wydajne metody selekcji cech z wykorzystaniem systemów uczących się, 13 października 2016
- [17] Wiesław Chmielnicki, Katarzyna Stąpor, Efektywne metody selekcji cech i rozwiązywania problemu wieloklasowego w nadzorowanej klasyfikacji danych, Kraków 2012

- [18] Joffrey L. Leevy, Taghi M. Khoshgoftaar, Richard A. Bauder, A survey on addressing high-class imbalance in big data, Nov 2018
- [19] Silva, D.A.; Souza, L.C.; Motta, G.H. An instance selection method for large datasets based on markov geometric diffusion. *Data Knowl. Eng.* 2016
- [20] Han, Wang, Mao, Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning. In *Proceedings of the International Conference on Intelligent Computing*, Hefei, China, 23–26 August 2005; Springer: Berlin/Heidelberg, Germany, 2005
- [21] Chawla, N.V.; Bowyer, K.W.; Hall, L.O.; Kegelmeyer, W.P. SMOTE: Synthetic minority over-sampling technique. *J. Artif. Intell. Res.* 2002
- [22] Yen S-J.; Lee Y-S. Cluster-based under-sampling approaches for imbalanced data distributions. *Expert Syst. Appl.* 2009
- [23] He H.; Bai Y.; Garcia E.; Li, S.A., Adaptive synthetic sampling approach for imbalanced learning. In *Proceedings of the IEEE International Joint Conference on Neural Networks*, 2008 (IEEE World Congress on Computational Intelligence), Hong Kong, China, 1–8 June 2008
- [24] Hempstalk, K.; Frank, E.; Witten, I.H. One-class classification by combining density and class probability estimation. In *Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, Antwerp, Belgium, 14–18 September 2008; Springer: Berlin/Heidelberg, Germany, 2008
- [25] Shin, Eom, Kim, One-class support vector machines—an application in machine fault detection and classification. *Comput. Ind. Eng.* 2005
- [26] Ertekin, S.; Huang, J.; Bottou, L.; Giles, L. Learning on the border: Active learning in imbalanced data classification. In *Proceedings of the Sixteenth ACM Conference on Information and Knowledge Management*, Lisbon, Portugal, 6–10 November 2007; ACM: New York, NY, USA, 2007
- [26] P.V. der Putten and M. van Someren, „A Bias-Variance Analysis of a Real World Learning Problem: The CoIL Challenge 2000,” *Machine Learning*, 2004
- [27] D. Mladenić and M. Grobelnik, „Feature Selection for Unbalanced Class Distribution and Naive Bayes,” *Proc. 16th Int’l Conf. Machine Learning*, 1999
- [28] G. Forman, „An Extensive Empirical Study of Feature Selection Metrics for Text Classification,” *J. Machine Learning Research*, 2003
- [29] Z. Zheng, X. Wu, and R. Srihari, „Feature Selection for Text Categorization on Imbalanced Data,” *ACM SIGKDD Explorations Newsletter*, 2004
- [30] Jacek Biesiada and Włodzisław Duch, *Feature Selection for High-Dimensional Data: A Pearson Redundancy Based Filter*
- [31] Rahul Agarwal, *Popularne metody selekcji cech w Data Science*, 09.2019
- [32] Younes Charfaoui, *Hands-on with Feature Selection Techniques: Filter Methods*, 01.2020

- [32] Witold Andrzejewski & Paweł Boiński, Drzewa Decyzyjne, Politechnika Poznańska, Wydział Informatyki
- [33] Krzysztof Krawiec, Wybrane zagadnienia uczenia maszynowego, Politechnika Poznańska, Wydział Informatyki
- [34] Girish Chandrashekar, Ferat Sahin, A survey on feature selection methods, Electrical and Microelectronic Engineering, Rochester Institute of Technology, Rochester, NY 14623, USA
- [35] Jose Manuel Pereira, Mario Bastoa Amelia, Ferreira da Silva, The Logistic Lasso and Ridge Regression in Predicting Corporate Failure, Procedia Economics and Finance Volume 39, 2016
- [36] L.E. Melkumova, S. Ya. Shatskikh, Comparing Ridge and LASSO estimators for data analysis, Procedia Engineering Volume 201, 2017
- [37] Hiroshi Motoda, Huan Li, Feature Selection Extraction and Construction, January 2002
- [38] Luis J. Mena, Jesus A. Gonzalez, Machine Learning for Imbalanced Datasets: Application in Medical Diagnostic, Conference: Proceedings of the Nineteenth International Florida Artificial Intelligence Research Society Conference, Melbourne Beach, Florida, USA, May 11-13, 2006
- [39] LHaibo He, Yang Bai, Edwardo A. Garcia, and Shutao Li, ADASYN: Adaptive Synthetic Sampling Approach for Imbalanced Learning
- [40] J R Quinlan, Bagging Boosting and C4.5, University of Sydney, Sydney Australia 2006
- [41] Tu Chengsheng, Liu Huacheng, Xu Bing, AdaBoost typical Algorithm and its application research, January 2017MATEC Web of Conferences
- [42] Mostafa Langarizadeh, Fateme Moghbeli, Applying Naive Bayesian Networks to Disease Prediction: a Systematic Review, 1Acta Inform Med. 2016 Oct
- [43] Taghi M. Khoshgoftaar, Moiz Golawala, Jason Van Hulse, Empirical Study of Learning from Imbalanced Data Using Random Forest, 19th IEEE International Conference on Tools with Artificial Intelligence(ICTAI 2007)
- [44] Francis Sahngun Nahm, Nonparametric statistical tests for the continuous data: the basic concept and the practical use, Feb 2016
- [45] Yoonsuh Jung, Multiple predicting K-fold cross-validation for model selection, Journal of Nonparametric Statistics, 2018
- [46] [a] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.KFold.html, 2.04.2020
- [47] Igor Kononenko, Edvard Simec, Marko Robin-Sik, Overcoming the myopia of inductive learning algorithms with RELIEFF, University of Ljubljana, Faculty of electrical engineering & computer science, Kluwer Academic Publishers, Boston
- [48] Ryan J. Meeker, Melissa LaCava, William Olson, Randal S. Moore, Jason H, Relief-Based Feature Selection: Introduction and Review, Journal of Biomedical Informatics, 2018

- [49] Kira Kenji and Rendell Larry, A Practical Approach to Feature Selection, Proceedings of the Ninth International Workshop on Machine Learning, 1992
- [50] Martin G. Larson, Analysis of Variance, Circulation, 2008
- [51] Mukesh Kumar Nitish, Kumar Rath Amitav Swain Santanu, Kumar Rath, Feature Selection and Classification of Microarray Data using MapReduce based ANOVA and K-Nearest Neighbor, Procedia Computer Science, Volume 54, 2015
- [52] Gongde GuoHui WangDavid BellYaxin BiKieran Greer, KNN Model-Based Approach in Classification, OTM 2003: On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE
- [53] [@] <https://www.kaggle.com/mlg-ulb/creditcardfraud>, 2.04.2020
- [54] [@] <https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.RobustScaler.html>, 3.04.2020
- [55] [@] https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.f_classif.html, 3.04.2020
- [56] [@] https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.chi2.html, 3.04.2020
- [57] [@] https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.mutual_info_classif.html, 3.04.2020
- [58] [@] <https://gitlab.com/moongoal/sklearn-relief>, 3.04.2020
- [59] [@] <https://www.geeksforgeeks.org/python-pandas-dataframe-corr/>, 3.04.2020
- [60] [@] <https://www.kaggle.com/c/porto-seguro-safe-driver-prediction>, 10.04.2020
- [61] [@] https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_classification.html, 11.04.2020
- [62] [@] <https://www.kaggle.com/anmolkumar/health-insurance-cross-sell-prediction>, 11.04.2020
- [63] [@] <https://www.kaggle.com/uciml/mushroom-classification>, 14.04.2020
- [63] [@] <https://sci2s.ugr.es/keel/imbalanced.php>, 14.04.2020
- [63] [@] <https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.LabelEncoder.html>, 13.04.2020

Zawartość płyty CD

Do pracy dołączono płytę CD o następującej zawartości:

- kod źródłowy programu znajdujący się w folderze /src
- otwartoźródłowe dane użyte do trenowania sieci neuronowych w katalogu /data
- katalog /docs zawierający kod źródłowy tej pracy
- katalog /results zawierający pliki wynikowe przeprowadzanych eksperymentów
- plik w formacie pdf zawierający tę pracę