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 inbalanced data classification problems.

AUTOR:
Maciej Hajduk

PROWADZĄCY PRACĘ:
dr inż. Paweł Trajdos, K02W04D03

WROCŁAW, 2021

Spis treści

Spis rysunków

Spis tablic

1 Wstęp

1.1 Wprowadzenie

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

The purpose of this paper was to compare different feature selection methods in the problem of training machine learning algorithms on imbalanced datasets. As part of it, currently popular selection methods were presented and described, and experiments were conducted for example datasets - both real and synthetic, with the goal of which was to create a ranking of algorithms. The author of the paper examined how properly performed selection affects the quality of results provided by a data classification program and how methods cited by him deal with data in which one class is significantly imbalanced. The engineering aspect consisted in the implementation of the experiments proposed in the paper, which allowed for a comprehensive comparison of algorithms.

Praca swoim zakresem objęła porównanie popularnych metod selekcji cech w ramach kilkuset wybranych zbiorów danych. Napisana w jej ramach biblioteka pozwoliła na stworzenie eksperymentów kompleksowo porównujących wyniki różnych algorytmów. Dane te posłużyć mogą do wyboru najodpowiedniejszej metody we wszelkich problemach klasyfikacyjnych, w których elementy wykazują szczególną nadreprezentację jednej bądź kilku klas. Zamieszczone w pracy podsumowanie zawiera wyniki przeprowadzonych przez autora badań.

Praca składa się z czterech rozdziałów:

Rozdział pierwszy: Omówiono analizę wybranego problemu, przedstawiono motywację podjęcia tematu oraz przegląd literatury. Opisano podstawowe metody selekcji cech i wyjaśnieniono różnicę pomiędzy selekcją oraz ekstrakcją cech. Zawarto opis szczegółowej charakterystyki zagadnienia i problemu jakim jest niezrównoważony rozkład klas w algorytmach uczenia maszynowego. Zamieszczono szczegółowy opis poszczególnych, wykorzystanych później metod selekcji, wraz z ich matematyczną interpretacją.

Rozdział drugi: Założenia i plan eksperymentu. W rozdziale zawarto informacje związane z inżynieryjnym aspektem pracy - czyli projekt systemu, szczegółowy plan poszczególnych ekspe-

rymentów i opis zbiorów danych, jakie użyto podczas doświadczeń. Swoim zakresem rozdział objął krótki opis użytych przez autora bibliotek oraz wykorzystywanych funkcji.

Rozdział trzeci: Podsumowanie uzyskanych wyników. W rozdziale zawarto rezultaty eksperymentów oraz testów opisanych w rozdziale drugim. Rozdział zawiera również opisy metod użytych w ramach porównywania rezultatów. Zamieszczone table i wykresy pozwalają na czytelną interpretacje danych.

Rozdział czwarty: W czwartym czwartym zawarto interpretację wyników oraz konfrontację ich z hipotezą postawioną na początku pracy. Wyjaśniono potencjalne zastosowanie przeprowadzonych w ramach pracy eksperymentów. Przedstawiono możliwości rozwoju projektu.

Udało się zrealizować wszystkie postawione cele.

2 Analiza problemu

Uczenie maszynowe to bardzo dynamicznie rozwijająca się gałąź informatyki. Niezwykły rozwój wynika z zapotrzebowania na wykrywanie prawidłowości, na uogólnianie oraz precyzowanie danych. Takie możliwości pozwoliły na znalezienie zastosowania dla algorytmów sztucznej inteligencji w wielu różnych branżach - począwszy od medycyny, poprzez finanse, produkcję i przemysł rozrywkowy. Tak duży przekrój różnych zastosowań wymaga ciągłego ulepszania istniejących już wzorców oraz tworzenia nowych, lepszych i bardziej efektywnych algorytmów. W większości praktycznych problemów klasyfikacji obiektów, autor operuje na dużej liczbie cech. Warto jednak pamiętać, że w tym przypadku "wiele" - nie oznacza lepszych rezultatów. Należy przytoczyć pojęcie "*przekleństwa wielowymiarowości*" [1]. Oznacza ono, że większy wymiar wymaga od programisty znacznie większej liczby danych, dodatkowo wraz ze wzrostem liczby cech wykładniczo rośnie liczba możliwych wariantów dopasowań, co znacznie zwiększa złożoność obliczeniową algorytmów.

Aby uniknąć problemów generowanych przez zbyt dużą liczbę atrybutów, a jednocześnie wykorzystać te, które zapewniają jak najlepszą separowalność klas, zazwyczaj pierwszym krokiem w zadaniu klasyfikacji jest selekcja lub ekstrakcja najodpowiedniejszych cech [2].

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 temat. Skutkuje to dużym przekrojem prac, również w ujęciu czysto dziedzinowym, jak wykorzystanie konkretnych algorytmów dla bardzo konkretnych zastosowań.

Wstępną analizę problemu przedstawili Garba Abdulrauf Sharifai i Zurinahni Zainol w pracy *Feature Selection for High-Dimensional and Imbalanced Biomedical Data Based on Robust Correlation Based Redundancy and Binary Grasshopper Optimization Algorithm* [3]. Autorzy opisali problem niezbalansowania danych i podejścia do niego, bazujące na istniejących metodach doboru cech. Skupili się szczególnie na zaproponowanej przez siebie metodzie mającej łaczyć różne metody filtrujące. Jest to dobre wprowadzenie do zadania selekcji cech i problemu radreprezentacji klas, a przytoczone metody są szczegółowo opisane co pozwala na dogłębne zrozumienie tematu. W podobnej pracy *Feature selection for high-dimensional imbalanced data*, twórcy - Liuzhi Yin, Yong Ge, Keli Xiao, Xuehua Wang i Xiaojun Quan [4] przyjrzeli się wpływowi, jaki na udaną selekcję ma zbiór danych. Postanowili przeprowadzić szereg eksperymentów, w szczególności dzieląc duże klasy na stosunkowo mniejsze pseudo-klasy i sztucznie generując ich etykiety. Na podstawie swoich badań zaprezentowali nową metodę selekcji opartą na tak zwanej *odległości Helingera*. Testy autorów pracy badają skuteczność tej metody na tle znanych już metod redukcji atrybutów. Twórcy pracy *A Survey on Evolutionary Computation Approaches to Feature Selection* [5] zajęli się przeglądem znanych metod tworząc dokument podsumowujący

każdą z nich, z jej wadami oraz zaletami. Punktem, na jakim szczególnie skupili swoją uwagę są nowoczesne techniki oparte o algorytmy ewolucyjne. Artykuł zawiera kompleksowy przegląd prac naukowych dotyczących tematu oraz a przytoczone badania pozwalają na dobranie odpowiedniej metody do zadanego zadania. W 2014 roku Deepika Tiwari opisała w artykule *Handling Class Imbalance Problem Using Feature Selection* [6] działanie kulku algorytmów wyboru cech dla niezrównoważonych zbiorów danych. W swoich eksprymentach skupiła się na uzyskaniu jak najlepszych wyników, co uzyskała modyfikując popularny algorytm RELIEFF, aby rozwiązać problem nierównowagi klas. W 2008 roku Chen Xuewen i Michael Wasikowski zaproponowali w swojej pracy *Combating the Small Sample Class Imbalance Problem Using Feature Selection* [7] porównanie różnych algorytmów stworzonych pierwotnie na potrzeby zadań klasyfikacji tekstu. Na uwagę zasługuje przytoczona przez autorów metoda nazwana *FAST*, która jest ich zdaniem doskonałym sposobem redukcji cech, zwłaszcza w przypadku zbiorów o niewielkiej ich liczbie. Mark A. Hall poruszył temat selekcji cech w swojej książce *Correlation-based Feature Selection for Machine Learning* [2], badając wartość zestawu cech na podstawie korelacji pomiędzy nimi. Autor przeprowadził szereg eksperymentów i porównał wymyśloną przez metodę opartą o współczynnik korelacji z metodami powszechnie stosowanymi, starając się między innymi wyodrębnić problemy, dla których jego algorytm jest najbardziej skuteczny. Analiza *CoIL Challenge 2000* [8] przeprowadzona przez Elkana Charlesa wykazała, że zwykłe algorytmy selekcji cech nie były wystarczająco dobre do zadania klasyfikacji danych niezbalansowanych. Stwierdził on, że na etapie selekcji należy rozważyć interakcję między różnymi cechami, a największą wadą, jaką znalazł w przypadku większości stosowanych metod selekcji cech, jest fakt, że nie rozważają wysoce skorelowanych cech. Isabelle Guyon i André Elisseeff przeprowadzili solidną analizę teoretyczną w pracy *An Introduction of Variable and Feature Selection* [9]. Wykazali w niej, 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.

Warto zwrócić uwagę, że niewiele jest prac naukowych badających skuteczność selekcji cech w zadaniu klasyfikacji danych niezbalanowanych. W szczególności, większość twórców skupia się na uzyskaniu jak najlepszych wyników, implementując w tworzone przez siebie algorytmy inne metody radzenia sobie z niezbalansowaniem, takie jak *oversampling* albo *boosting*. Autorzy artykułów posługują się w badaniach niewielką liczbą specjalnie przystosowanych zbiorów, co nie pozwala na obiektywne i kompleksowe porównanie różnych metod. Motywacją tej pracy było uzupełnienie istniejących w aktualnym stanie wiedzy luk. W ramach testów zbadane zostały przytoczone w analizie literatury algorytmy, a duża liczba zróżnicowanych zbiorów danych pozwoliła na otrzymanie prawidłowych wyników.

2.2 Cel selekcji cech

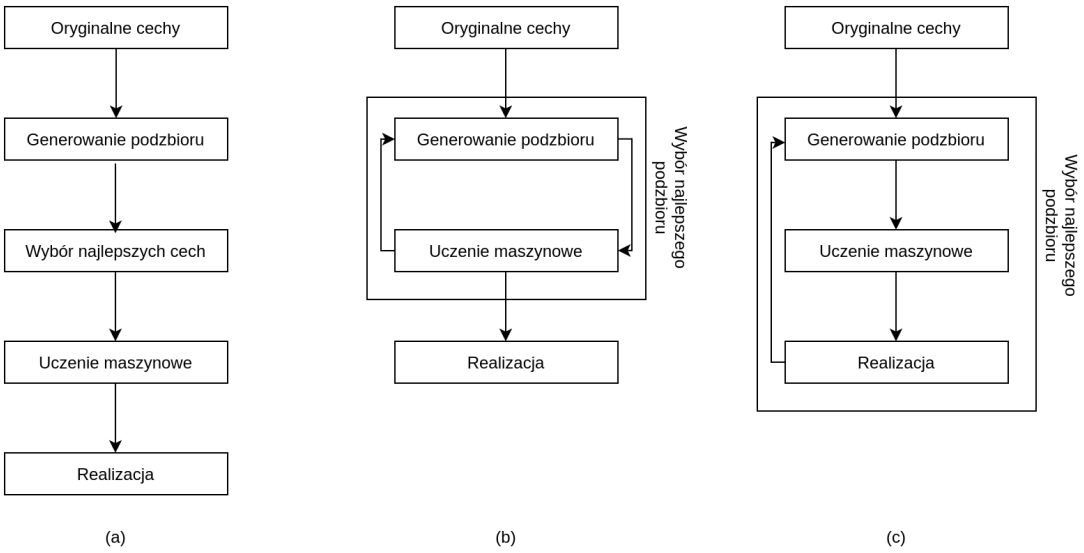
Selekcja cech polega na identyfikacji elementów puli atrybutów, które uznawane są za najlepsze deskryptory rozważanych kategorii. Zaletą selekcji jest możliwość zbadania tych deskryptorów, które są istotne z punktu widzenia danego zadania klasyfikacji, czyli jednocześnie zrozumienie różnic między analizowanymi kategoriami. Wybór najbardziej istotnych atrybutów badanych

obiektów przekłada się bezpośrednio na poprawne działanie klasyfikatora [2]. Dyspozycja coraz większymi bazami danych zmusza do optymalizacji 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” [10]. Zjawisko to zachodzi zazwyczaj, gdy liczba cech znacznie przewyższa liczebność samego zbioru danych. Zadaniem selekcji jest również lepsze zrozumienie problemu oraz zmniejszenie kosztów archiwizacji przyszłych danych.

W kolejnych rozdziałach opisano trzy główne metody tworzenia algorytmów selekcji: metody rankingowe - zwane filtrami, metody opakowane oraz metody wbudowane [11]. Dla każdej z wymienionych metod została nakreślona idea, oraz przedstawione zostały algorytmy reprezentujące daną metodologie [2][9][10].

2.3 Podstawowy podział

W rozdziałach 2.3.1 - 2.3.3 opisano trzy główne metody tworzenia algorytmów selekcji. Podział ten i działanie każdej z metod obrazuje rysunek 2.1.



Rysunek 2.1: Trzy główne podejścia do problemu selekcji cech - metody rankingowe (a), metody opakowane (b) oraz metody wbudowane (c).

2.3.1 Metody rankingowe

Najprostsze podejście do problemu selekcji cech reprezentowane jest poprzez metody rankingowe, nazywane też filtrami [11]. Jak sama nazwa wskazuje, do zadania selekcji przy pomocy metod rankingowych należy podejść wyróżniając w zbiorze cech następujące grupy: cechy

istotne, nieistotne i redundantne. Istotne to takie, które odróżniają od siebie klasy, nieistotne nie niosą informacji dla problemu klasyfikacji, a cechy redundantne to atrybuty których role z powodzeniem mogą przejąć inne cechy.

Metody rankingowe polegają więc na znalezieniu pewnej miary pozwalającej stworzyć ranking cech, a następnie wybrać najlepsze i odrzucić najgorsze atrybuty. Algorytmy te zazwyczaj są najszybsze i - co istotne - nie zależą one od używanej metody analizy danych [11]. Ich istotną wadę stanowi brak możliwości uwzględnienia zależności pomiędzy cechami [6]. Kolejne opisane typy metod selekcji tej wady nie posiadają. Istnieje wiele popularnych filtrów, wśród których wyróżnić można najbardziej popularne - *chi-kwadrat*, *ANOVA* oraz *RELIEF* [6]. Podejście rankingowe do selekcji cech zostało zobrazowane na rysunku 2.1, w podpunkcie (a).

2.3.2 Metody opakowane

Podstawowymi i najpopularniejszymi metodami selekcji cech są metody opakowane, tak zwane wrappery [11]. W przeciwieństwie do metod rankingowych, w których algorytm selekcji i klasyfikator pozostają niezależne, w algorytmach opakowanych, ocena atrybutów dokonuje się przy użyciu konkretnego modelu. To właśnie efektywność samego klasyfikatora służy za miarę skuteczności algorytmu. Zaletą metody jest jej uniwersalność i dokładność, natomiast wadą - wysoka złożoność obliczeniowa [11]. Dla efektywności tych algorytmów istotny jest sposób ustalania podzbioru cech [11]. Wśród wielu sposobów wyszukiwania właściwego podzbioru, wyróżnić można najprostszy - przeszukanie całego zbioru podzbiorów. Jest to jednak rozwiązanie bardzo kosztowne. Wobec tego typowymi strategiami są: przeszukiwanie w przód, przeszukiwanie wstecz oraz tworzenie indywidualnego rankingu [12]. Popularnym algorytmem implementującym metodę opakowaną jest *RFE - Rekurencyjna eliminacja cech* [11]. Podejście opakowane do selekcji cech zostało zobrazowane na rysunku 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 [5][13][14]. W literaturze natknąć się też można na przypasowanie do tej kategorii metody wektorów nośnych (SVM) czy też analizy składowych głównych (PCA) [5]. Zaletą tych metod jest ich szybkość, ponieważ użycie nie wiąże się z dodatkowymi operacjami na zbiorze [11][5][14]. Podejście wbudowane do selekcji cech zostało zobrazowane na rysunku 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

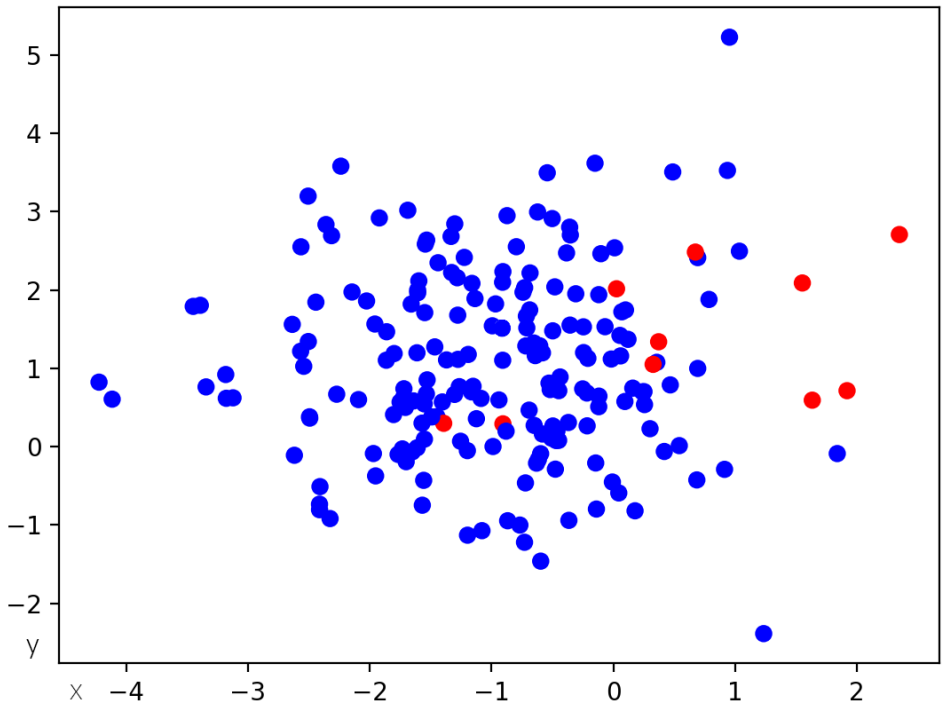
danych [15]. Zawsze rozważana jest ona w kontekście kolejnych zadań i nie można oceniać jej skuteczności w oderwaniu od wyników metody klasyfikacji wykorzystującej wybrane cechy. W większości przypadków budowany jest złożony model, który może zawierać jeden lub więcej algorytmów selekcji i co najmniej 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, równym lub nawet większym od wymiaru przestrzeni startowej [16][15]. Poprzez proces ekstrakcji, część początkowych cech zostaje utracona. Z tego powodu nie ekstrakcja, a selekcja cech jest obecnie najpowszechniejszą strategią służącą przygotowaniu reprezentacji analizowanych danych [16].

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. Postęp w dziedzinie sztucznej inteligencji i metodach statystycznych stworzyły nowe możliwości klasyfikacji i diagnozy najbardziej złożonych i śmiertelnych chorób, takich jak rak, choroba Alzheimera, cukrzyca itp, których wykrycie w początkowych stadiach stanowi problem [3]. Między innymi z takimi przypadkami wiąże się problem niezrównoważonej dystrybucji klas [17].

Niezrównoważony rozkład ma miejsce, gdy co najmniej jedna z klas jest niewystarczająco reprezentowana i przytłoczona przez inne klasy. Algorytmy klasyfikacji często nie radzą sobie z niezrównoważonymi danymi, co stwarza wiele przeszkód w uczeniu się algorytmów i implikuje liczne konsekwencje dla rzeczywistych zastosowań. Problem objawia się niedocenianiem 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 takiej sytuacji zaobserwować można na rysunku 2.2. Ilustracja przedstawia 200 elementów z których tylko 5% należy do klasy mniejszościowej - czerwonej.



Rysunek 2.2: Przykład niezrównoważonego rozkładu klas.

Sytuacja przedstawiona na grafice jest dla algorytmu niewygodna, ponieważ większość tradycyjnych algorytmów uczenia maszynowego trenowana na podobnym zbiorze, obciążona zostanie w stosunku do klasy bardziej licznej [18]. Jednocześnie, zazwyczaj lepsze zrozumienie klas mniej licznych jest istotniejsze z punktu widzenia problemu w ujęciu biznesowym [17]. Problemem jest również określenie wartości 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 w tym przypadku nie rozpoznawać żadnego elementu klasy mniejszościowej [18].

2.6 Metody klasyfikacji danych niezbalansowanych

Problem nierównoważnego rozkładu przyciągnął w ostatnim czasie zainteresowanie dużej części społeczności zajmującej się uczeniem maszynowym i eksploracją danych, zarówno ze środowisk akademickich jak i środowisk związanych z przemysłem. Znajduje to odbicie w dużej liczbie startupów opierających swoje produkty i usługi na rozwiązaniach *machine-learningowych*. W ciągu kilkunastu ostatnich lat wyklarowały się trzy główne podejścia do uczenia modeli na danych niezbalansowanych. Są to metody na poziomie danych, metody na poziomie algorytmów oraz metody hybrydowe [19][20][21][22].

2.6.1 Metody na poziomie danych

Metody na poziomie danych (Data-level methods), modyfikują dostępne instancje problemu w celu jego zbalansowania. Można je podzielić na podgrupy: metody próbkowania danych (data-sampling) i metody wyboru cech (feature selection methods) [23].

Metody nadpróbkowania i podpróbkowania stanowią dwie podgrupy metod próbkowania danych, w których próbkowanie dla danego zbioru danych odbywa się losowo lub z wykorzystaniem innego, zdefiniowanego wcześniej algorytmu. W procesie nadpróbkowania (oversamplingu) do danego zbioru danych dodawane są instancje klasy mniejszościowej poprzez replikację, która odbywa się z wykorzystaniem algorytmów takich jak ADASYN [22]. W procesie podpróbkowania (undersamplingu) natomiast, większość wystąpień klasy zostanie usuniętych z danego zbioru danych. 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 [20]. Chociaż metoda może skutkować znacznie lepszymi wynikami w porównaniu z oryginalnym zestawem danych, istnieją poważne problemy związane z jej wykorzystaniem po pobraniu zbyt małej liczby próbek. Wydajność klasyfikatora może wtedy ulec pogorszeniu z powodu potencjalnej utraty przydatnych przykładów klasy większościowej [1]. 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 [23].

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 w ramach klasyfikacji. Zmniejsza to niekorzystny wpływ nierównowagi klas na wyniki klasyfikacji [23].

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 methods) i metody zintegrowane (integrated methods) [23].

Pierwsza z nich opiera się na zasadzie przypisywania większej wagi instancjom w przypadku błędnej klasyfikacji. Na przykład fałszywie negatywnym przewidywaniom można przypisać wyższy koszt niż fałszywie dodatnim.

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 [23]. Dwa powszechne typy metod uczenia zintegrowanego to Bagging i Boosting [24]. Bagging minimalizuje wariancję, generując kilka zestawów uczących dla zestawu

danych i tworząc klasyfikator dla każdego zestawu uczącego, a następnie łącząc ich modele w celu ostatecznej klasyfikacji. Algorytmy wykorzystujące Boosting, takie jak algorytm *AdaBoost*, tworzą serię klasyfikatorów, wszystkie stosowane na tym samym zestawie danych [25]. *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. Próbka, 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 identyfikacje obiektu jednej, określonej klasy, poprzez uczenie się przede wszystkim ze zbioru zawierającego tylko obiekty tej klasy, mają na celu zwalczenie problemu nadmiernego dopasowania. Występuje on przypadku większości klasyfikatorów uczących się na niezrównoważonych danych. Osiągane jest to poprzez podejście do tego problemu z pomocą uczenia nienadzorowanego [26][27][28]. 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 [29]). W niektórych przypadkach podgrupy metod na poziomie danych lub podgrupy metod na poziomie algorytmu można łączyć jako ogólną metodę rozwiązywania problemu niezbalansowania klas. Na przykład popularny klasyfikator losowego lasu (Random Forest) jest wersją oryginalnego algorytmu losowego lasu decyzyjnego (Random Decision Forest) i jest zintegrowany z algorytmem uczenia się, który dodatkowo implementuje Bagging [30].

2.7 Metody oparte o selekcję cech

Pojęcie przekleństwo wymiarowości mówi, że jeśli wiele cech jest zaszumionych, czyli wprowadza do zbioru dane redundantne lub nawet błędne, koszt użycia klasyfikatora może być bardzo wysoki, a wydajność może być poważnie zaniżona [1]. Ponieważ problemowi z nierównowagą klas często towarzyszy problem dużej wymiarowości zbioru danych, zastosowanie technik selekcji cech jest koniecznym działaniem [1]. 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 pomógł w walce z problemem nadmiernego dopasowania [31]. Forman odnotował podobną obserwację dotyczącą wysoce niezrównoważonych problemów klasyfikacji tekstu i stwierdził, że „żaden stopień sprytnej indukcji nie może zrekompensować braku sygnału predykcyjnego w przestrzeni wejściowej" [32][33][34]. Badania te pokazują, że w wielowymiarowych zbiorach danych, sam dobór cech może zwalczyć problem nierównowagi klas.

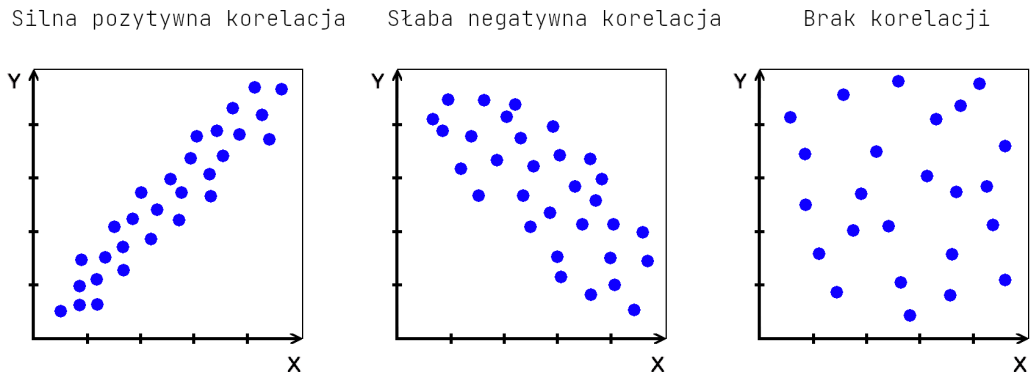
W ostatnich latach, radzenie sobie z niezrównoważonymi zbiorami danych za pomocą selekcji cech stało się popularne wśród społeczności zajmujących się eksploracją danych i uczeniem maszynowym [8]. Wspomniane wcześniej techniki koncentrują się na próbkowaniu danych uczących w celu przezwyciężenia niezrównoważonego 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 [35]. 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. Pozwala 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 poniekąd 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 drugiej. W przypadku liniowych modeli uczenia maszynowego, częstym celem będzie znalezienie elementów silnie skorelowanych ze zmienną losową opisującą przynależność do klasy. Jednocześnie, dwie silnie skorelowane ze sobą zmienne dostarczają też redundantnych informacji. Zasadniczo można dokonać poprawnego sklasyfikowania z pomocą tylko jednego z tych atrybutów. Usunięcie drugiego może wręcz 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 [36]. Znak współczynnika wskazuje kierunek zależności: znak dodatni informuje, że dwie zmienne rosną lub maleją względem siebie (pod względem korelacji), a znak ujemny wskazuje, że jedna zmienna rośnie, a druga maleje [35]. Interpretacje wyników korelacji na przykładowych zbiorach danych zostały przedstawione na rysunku 2.3.



Rysunek 2.3: Interpretacje współczynnika korelacji dla różnych zbiorów danych.

Ogólna postać współczynnika przedstawiona została w równaniu 2.1:

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

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 zmiennym obliczając macierz relacji między każdą parą zmiennych w zbiorze danych. Rezultatem jest symetryczna macierz zwana macierzą korelacji.

2.7.2 Chi-square

Chi-square (chi-kwadrat lub χ^2) jest testem statystycznym mierzącym niezależność cechy od etykiety klasy. Mierzy on zależność między zmiennymi stochastycznymi. Uużycie 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 [36].

Ogólna postać testu została przedstawiona w równaniu 2.2:

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

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 [33]. Chociaż test chi-kwadrat zadowalająco uogólnia dane dyskretne, nie radzi sobie dobrze się podczas testowania danych ciągłych [6].

2.7.3 Information Gain

Entropia warunkowa to entropia po podziale zbioru przy pomocy danego atrybutu [6]. Dla danego atrybutu a, entropia warunkowa wyraża się wzorem 2.3:

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

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:

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

gdzie:

- p_i to prawdopodobieństwo przynależności do klasy $i - tej$,
- S to zbiór przykładów,
- k liczba klas.

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 [6]. Metoda ta ocenia przyrost informacji przy użyciu atrybutu. Dla danego atrybutu a:

$$IG(S, a) = Ent(S) - Ent(S|a) \tag{2.5}$$

Podobnie jak test *Chi-square*, metoda *Information Gain* dobrze uogólnia atrybuty dyskretne, ale nie radzi sobie z atrybutami danych ciągłych. Ponadto preferuje atrybuty o dużej liczbie wartości i może prowadzić do przeuczenia [32]. Problemy te rozwiązuje zmodyfikowana wersja algorytmu - *Gain Ratio*, wykorzystująca taka zwaną *wartość wewnętrzną*, która koryguje obciążenie poprzez dostarczenie danych o wielkości zbioru.

2.7.4 Relief i ReliefF

Grupa algorytmów *Relief* jest jednym z najskuteczniejszych przedstawicieli opracowanych dotychczas metod filtrujących [6]. Większość metod filtrowania opracowanych w celach eksploracji danych i uczenia maszynowego zakłada warunkową niezależność atrybutów. Algorytmy Relief są pod tym względem rewolucyjne, 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 [6].

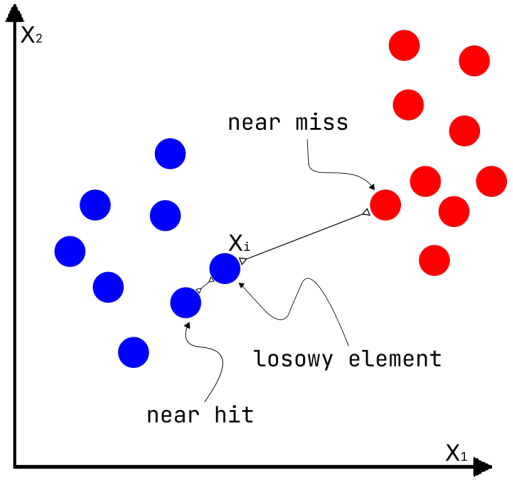
Podstawową ideą algorytmów Relief jest oszacowanie jakości cech na podstawie tego, jak dobrze cecha może rozróżniać instancje, które są blisko siebie. Relief oblicza punktację dla każdej cechy, którą można następnie zastosować do selekcji cech. Punktacja opiera się na wyliczeniu różnic wartości cechy 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 [37]. Algorytm został pierwotnie zaprojektowany do zastosowań w problemach klasyfikacji binarnej. Relief zainspirował rodzinę algorytmów wyboru cech - RBA (Relief Based Alghoritms), w tym metodę ReliefF [37]. Jest ona dodatkowo przystosowana 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 liczbie 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:

$$W_i = W_i - (x_i - nearHit_i)^2 + (x_i - nearMiss_i)^2, \tag{2.6}$$

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. W taki sposób, tworzy on ranking cech [37].

Zaletą metod RBA jest to, że nie są one zależne od heurystyki, działają w czasie wielomianowym niskiego rzędu, są odporne na zakłócenia, a także nadają się do danych binarnych lub ciągłych [38][39].

2.7.5 ANOVA

ANOVA pochodzi od angielskiego “analysis of variance,” czyli analiza wariancji. Jest to metoda, która wyjaśnia, z jakim prawdopodobieństwem wyodrębnione czynniki mogą być powodem różnic między obserwowanymi średnimi grupowymi. Algorytm polega na porównaniu wariancji międzygrupowej do wariancji wewnątrzgrupowej. ANOVA to jeden z algorytmów statystyki F (F-test) - od nazwiska twórcy, Ronalda Fishera [40].

Analizę wariancji można podzielić na trzy grupy analiz:

- **jednoczynnikowa analiza wariancji** - wpływ jednego czynnika międzygrupowego na zmienną zależną
- **wieloczynnikowa analiza wariancji** - wpływ kliku 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 wariancja przypadająca na każdy z analizowanych czynników jak również wariancja błędu. Idealna sytuacja ma

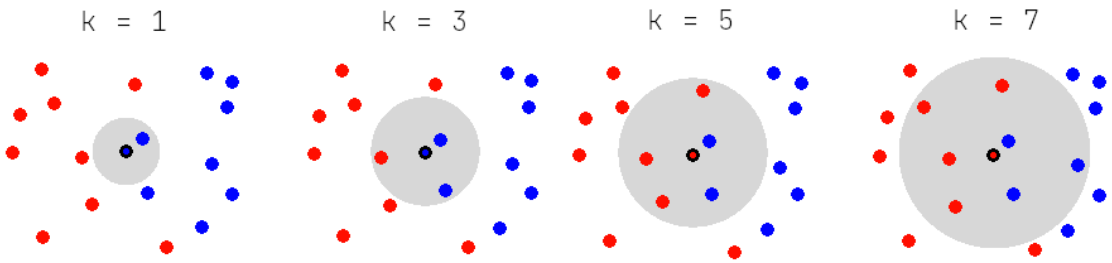
miejsce wtedy, gdy wariancja międzygrupowa jest duża, natomiast wariancja wewnątrzgrupowa - mała [40][41].

3 Założenia i plan eksperymentu

Obiektywne porównanie przytoczonych algorytmów cech wymagało przeprowadzenia szeregu eksperymentów porównujących skuteczność popularnych metod selekcji cech. Praca swoim zakresem objęła eksperymenty przeprowadzone na kilkuset wybranych zbiorach danych. Użyto zarówno zbiorów rzeczywistych - to znaczy zebranych w ramach rzeczywistych pomiarów jak i syntetycznych - wygenerowanych przez algorytm. Bazy danych, użyte w ramach badań implikują skupienie się na problemach dwuklasowych. Hipoteza, z którą twórca konfrontuje wyniki eksperymentów, to założenie 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 jezyk Python w wersji `3.8` oraz biblioteki `sckit-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 atrybutów, które będą używane przez algorytm w celu przeprowadzenia klasyfikacji. W celu 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, badany obiekt przydzielany jest do klasy, do której należy większość z jego sąsiadów [40][42].



Rysunek 3.1: Graficzna reprezentacja działania algorytmu KNN - przydzielenie badanej instancji do jednej z dwóch klas na podstawie jej sąsiadów. Obiekt może zostać oetykietowany inaczej w zależności od parametrów, w tym przypadku promienia obszaru.

Algorytm przyjmuje zbiór danych uczących zawierający elementy. Każdy element 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 stanowi w badanych przypadkach problem. Dokładność (accuracy) używana jako metryka ewaluacji może być w takim wypadku niewystarczająca, gdyż nawet model o skuteczności 95% - co jest na ogół wartością bardzo dobrą - w przypadku rozkładu 5/95 mógłby nie rozpoznawać żadnego elementu klasy mniejszościowej. Metrykami, które dostarczą bardziej wartościowe dane są:

- Macierz konfuzji: tabela pokazująca prognozy prawidłowe i nieprawidłowe z podziałem na klasyfikacje pozytywne i negatywne.
- 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:

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

Na podstawie macierzy konfuzji została obliczona także krzywa ROC (ang. Receiver Operating Characteristic), która opisuje zależność między czułością (sensitivity), a dokładnością (specificity) modelu. Czułość i dokładność będą wyznaczane kolejno przy pomocy wzorów:

$$TPR = \frac{TP}{TP + FN}, \tag{3.2}$$

gdzie TP to liczba wyników *True Positive*, a FN to liczba wyników *False Negative* w macierzy konfuzji.

$$FPR = \frac{FP}{TN + FP}, \tag{3.3}$$

gdzie FP to liczba wyników *False Positive*, a TN to liczba wyników *True Negative* w macierzy konfuzji.

Z pomocą tych wartości wylicza się również tak zwana dokładność zbalansowana (balanced accuracy). Normalizuje ona przawdziwie pozytywne i prawdziwie negatywne prognozy odpowienio przez liczbę próbek dodatnich i ujemnych i dzieli ich sumę przez dwa:

$$bACC = \frac{TPR + TNR}{2}, \tag{3.4}$$

gdzie TNR oznacza *True Negative Rate* i jest równe $1 - FPR$.

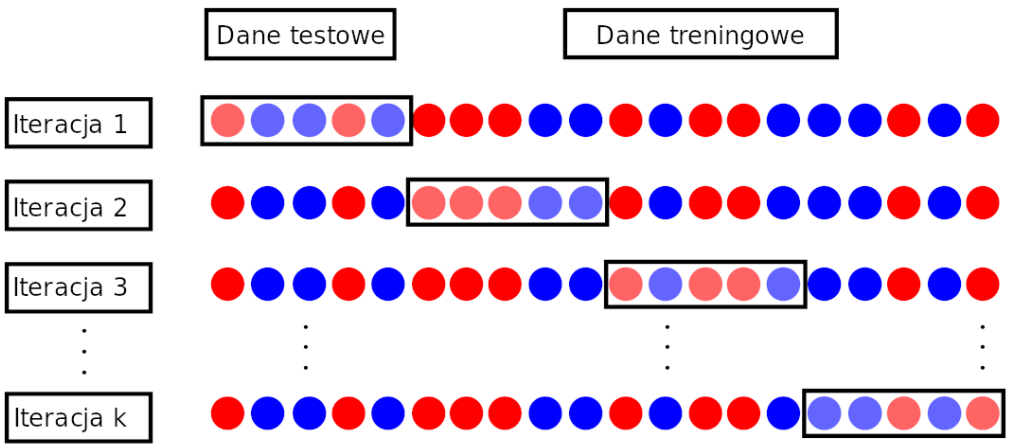
Pod uwagę wzięty został również ranking cech wygenerowany przez poszczególne metody na poszczególnych zbiorach. W celu określenia, która z testowanych metod daje najlepsze wyniki klasyfikacji wykorzystany zostanie test statystyczny - test Wilcoxona [43]. Do jego wykonania użyto wartości *F1 Score* uzyskanych 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 podzbiory: treningowy i testowy. Jest to konieczne w celu wydzielenia 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.

Istnieje kilka metod podziału zbioru. Najprostszym 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, zawyżoną dokładność. 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*, czyli tak zwana *walidacja krzyżowa* [44].

W algorytmie walidacji krzyżowej (rys. 3.2), zbiór dzielony jest losowo na k równych podzbiorów. Następnie kolejno każdy z nich używany jest jako zbiór testowy, a połączona reszta - 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* [45]. Podobnie jak inne funkcje z tej biblioteki, umożliwia ona zadeklarowanie ziarna losowości, co zapewnia możliwość powtórzenia użyskanych w ten sposób wyników.

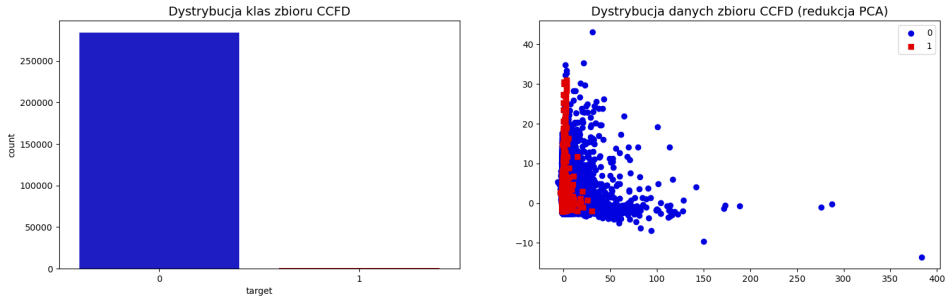
3.4 Zbiory danych

W ramach przeprowadzonych doświadczeń posłużono się 115 zbiorami danych. Wszystkie charakteryzowały się wysokim niezbalansowaniem i wykazywały nadreprezentację jednej z klas. Rozdziały 3.4.1 - 3.4.4 zawierają szczegółowy opis dużych, starannie przygotowanych w ramach pracy zbiorów z różnych dziedzin nauki. Bazy te prezentują najbardziej typowe przypadki wykorzystania różnych metod selekcji cech. Rozdział 3.4.5 skrótowo opisuje pozostałe zbiory danych, uzyskane za pośrednictwem platformy *KEEL*.

3.4.1 Wykrywanie oszustw kredytowych

Zbiór danych “Credit Card Fraud Detection” (*CCFD*) zawiera informacje o transakcjach dokonanych kartami kredytowymi we wrześniu 2013 roku, przez europejskich posiadaczy kart. Baza składa się z zapisów transakcji, które odbyły się ciągu dwóch dni, w trakcie których miały miejsce 492 oszustwa z 284 807 wszystkich operacji. Zbiór jest wysoce niezbalansowany, a klasa pozytywna (oszustwa) stanowi 0,172% wszystkich transakcji [46].

Elementy zbioru składają się jedynie z danych liczbowych, które są wynikiem transformacji PCA. Jest to podyktowane względami poufności - twórcy nie są w stanie dostarczyć oryginalnych wartości ani dodatkowych informacji o danych. Cechy $V1, V2, \ldots V28$ są głównymi składnikami uzyskanymi za pomocą analizy składowych głównych, jedynymi cechami, które nie zostały przekształcone są “*Time*” i “*Amount*.” Warość “*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*.



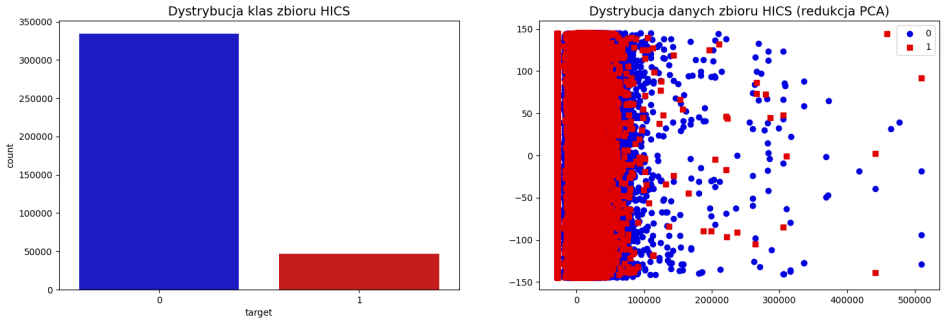
Rysunek 3.3: Dystrybucja klas i danych dla zbioru Credit Card Fraud Detection.

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

3.4.2 Ubezpiecznia zdrowotne

Zbiór "Health Insurance Cross Sell" (*HICS*) został pierwotnie stworzony w ramach konkursu. Celem wyzwania było przewidywanie prawdopodobieństwa, że klienci firmy ubezpieczeniowej będą zainteresowani ubezpieczeniem komunikacyjnym oferowanym przez firmę [47]. Każdy wiersz odpowiada określonemu posiadaczowi polisy, a kolumny opisują ich cechy. Zmienna docelowa jest tu 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.

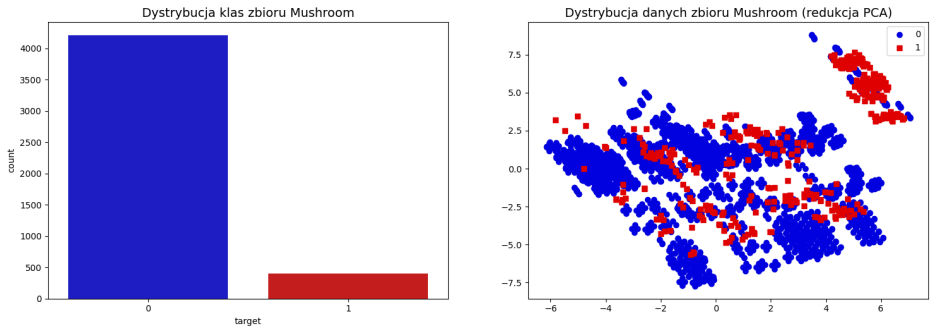
Zbiór został pozyskany za pośrednictwem platformy `Kaggle` (`https://www.kaggle.com/`).

3.4.3 Klasyfikacja grzybów

Zestaw danych "Mushroom Classification" zawiera opisy próbek różnych gatunków grzybów, zaczerpnięte z The Audubon Society Field Guide to North American Mushrooms (1981) [48]. Każdy gatunek jest określony jako zdecydowanie jadalny, zdecydowanie trujący lub o nieznanej jadalności (niezalecany). Ta ostatnia klasa została połączona z klasą trującą. Przewodnik jasno stwierdza, że nie ma prostej zasady określania jadalności grzyba co powinno uniemożliwić skuteczną klasyfikację. 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 itd. Wszystkie cechy opisane są zmiennymi

tekstowymi. Sam zbiór nie wykazuje cech niezbalansowania. Został odpowiednio dostosowany do celów eksperymentu. Proces ten opisano 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*.

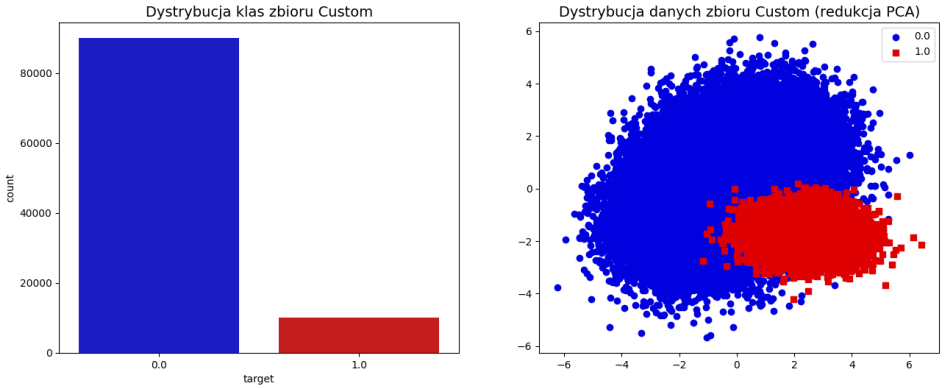


Rysunek 3.5: Dystrybucja klas i danych dla zbioru Mushroom.

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

3.4.4 Dane wygenerowane syntetycznie

Zbiór "Custom" został wygenerowany syntetycznie z pomocą funkcji *make_classification* biblioteki *scikit-learn* [49]. Pozwoliło to na dobranie parametrów w taki sposób, aby zbiór odpowiadał jak najlepiej założeniom projektowym. Baza 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 Pozostałe zbiory danych

W celu poprawnego i obiektywnego porównania wszystkich metod selekcji wykonano eksperymenty na dodatkowych 111 zbiorach udostępnionych przez platformę KEEL [50]. 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 w większości cechy opisane numerycznie i jedynie dwie klasy - opisane za pomocą etykiet "*positive*" oraz "*negative*." Ponadto, klasą mniejszościową jest zawsze klasa "*positive*."

3.5 Przygotowanie danych

Modele predykcyjne wymagają zazwyczaj danych wysokiej jakości. 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,
- zakłócenia: dane zawierają błędne rekordy lub elementy odstające,
- niespójność: dane zawierają rekordy powodujące niezgodności.

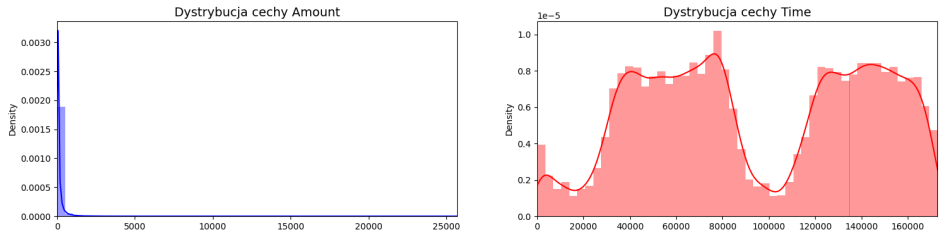
W celu odpowiedniego, wstępnego przetworzenia danych, do każdego zbioru należy podejść indywidualnie. Są jednak metody, które pozwalają poradzić sobie z większością napotkanych problemów. Do popularnych metod radzenia sobie z problemami wynikającymi ze źle przygotowanymi bazami danych należą:

- uzupełnienie 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. iloczyny 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 Przygotowanie 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 rysunku 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* [51].

3.5.2 Przygotowanie zbioru HICS

W zbiorze *Health Insurance Cross Sell* nie występują puste wartości. Większość cech to cechy binarne, które nie potrzebują obróbki. 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 przepisane na dane binarne: "*0*" i "*1*." Kolumna opisująca wiek pojazdu posiada 3 możliwe wartości tekstowe, które zostały przepisane na wartości liczbowe: "*0*," "*1*" i "*2*."

3.5.3 Przygotowanie zbioru Mushroom

Zbiór *Mushroom* był początkowo zbiorem zbalansowanym. W celu niezbalansowania go, usunięto 90% elementów zakwalifikowanych jako grzyby trujące (zostawiając co dziesiąty element oznaczony klasą nawaną '*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* z biblioteki *scikit-learn* [52]. Metoda ta podmienia każdy element listy na wartość pomiędzy 0 a n-1, gdzie n to liczba wariantów cechy.

3.5.4 Przygotowanie zbioru Custom

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

3.5.5 Przygotowanie zbiorów biblioteki KEEL

Zbiory z bazy danych *KEEL* zostały przez twórców przygotowane do testów, bez konieczności uprzedniego nadmiernego poprawiania danych. Z pośród 111 zbiorów, istniało jedynie kilkanaście, zawierających wartości cech w formacie tekstowym. Funkcja przygotowująca sprawdziła każdy zbiór pod kątem takich przypadków i w razie potrzeby podmieniła takie atrybuty na dane numeryczne. Użyta została w tym celu metoda *LabelEncoder* z biblioteki *scikit-learn* [52]. Preprocessing pozostałych zbiorów ograniczył się do podmiany wartości "*positive*" oraz "*negative*" na wartości "*1*" i "*0*" na etapie parsowania zbioru.

3.6 Eksperymenty

W celu wykonania analizy, przy pomocy bibliotek *scikit-learn* oraz *pandas*, zaimplementowano metody opisane w rozdziałach 2.7.1 - 2.7.5. Algorytmy *ANOVA*, chi^2 oraz *Information Gain* mają gotową implementację w bibliotece *scikit-learn* - kolejno funkcje *f_classif* [53], *chi2* [54] i *mutual_info_classif* [55]. Zostały one wykorzystane w ramach funkcji *SelectKBest*, umożliwiającej zdefiniowanie metody selekcji cech oraz docelowej liczby atrybutów. Algorytm *Relieff* wdrożony został przy użyciu biblioteki *scikit-feature* [56][57]. Metoda *Correlation Coefficient* zrealizowana została przy użyciu funkcji *corr* z pakietu *pandas*, zwracającej korelację Pearsona pomiędzy parami wszystkich kolumn danego zbioru [58]. Obie funkcje dostosowano tak, by odpowiadały swoim działaniem oraz API pozostałym trzem metodom. Eksperymenty zostały powtórzone wiele razy z różnymi ustawieniami w celu ustalenia optymalnej liczby cech dla każdego zbioru.

W celu ułatwienia sprawnego i obiektywnego porównania wszystkich metod, w ramach pracy stworzono bibliotekę zawierającą fukcje przygotowujące dane, dzielące zbiór na podzbiory (walidacja krzyżowa), przeprowadzające klasyfikacje oraz wyliczające i prezentujące wyniki eksperymentów. Rezultatem każdego doświadczenia jest plik w formacie *.csv*. Plik ten zawiera nazwę zbioru użytego do klasyfikacji, nazwę metody selekcji, sumaryczną liczbę cech oraz liczbę cech po przeprowadzonej selekcji, dokładność, zbalansowaną dokładność, precyzję, czułość, wynik testu *F1 Score*, wartość *FPR*, *TPR* i *TNR*, macierz konfuzji i cechy wybrane w ramach selekcji w 6 wariantach - przed selekcją i dla każdej użytej metody.

dataset	method	num_of_feat	num_of_elems	accuracy	balanced_acc	precision	recall	f1_score	tpr_fpr_tnr
ecoli-0_vs_1.dat	NO SELECTION	7	220	0.9645454545454546	0.97413793103448275	1.0	0.9482758620689654	0.9734513274336283	(0.9482758620689654,
ecoli-0_vs_1.dat	ANOVA	3	220	0.9727272727272727	0.9768512887861519	0.9944055944055943	0.9647218453188603	0.9793388429752065	(0.9647218453188603,
ecoli-0_vs_1.dat	RELIEF	3	220	0.9709090909090909	0.9785809906291834	1.0	0.9571619812583667	0.9781121751025992	(0.9571619812583667,
ecoli-0_vs_1.dat	INFORATION GAIN	3	220	0.9681818181818181	0.9766666666666667	1.0	0.9533333333333334	0.9761092150170648	(0.9533333333333334,
ecoli-0_vs_1.dat	CHI SQUARE	3	220	0.9709090909090909	0.9747954462662718	0.993086993086993	0.9633649932157394	0.977961432506887	(0.9633649932157394,
ecoli-0_vs_1.dat	CORRELATION COEF	3	220	0.9727272727272727	0.9768512887861519	0.9944055944055943	0.9647218453188603	0.9793388429752065	(0.9647218453188603,
ecoli1.dat	NO SELECTION	7	336	0.8869047619047619	0.8545028529970728	0.6727272727272727	0.8018575851393189	0.731638418079096	(0.8018575851393189,
ecoli1.dat	ANOVA	3	336	0.8482142857142858	0.785252700617284	0.6675324675324675	0.6692708333333334	0.6684005201560468	(0.6692708333333334,
ecoli1.dat	RELIEF	3	336	0.8482142857142858	0.785252700617284	0.6675324675324675	0.6692708333333334	0.6684005201560468	(0.6692708333333334,
ecoli1.dat	INFORATION GAIN	3	336	0.8898809523809523	0.8661451958384333	0.6545454545454545	0.8289473684210527	0.7314949201741655	(0.8289473684210527,
ecoli1.dat	CHI SQUARE	3	336	0.8517857142857143	0.7887545555756332	0.7116883116883117	0.6650485436893203	0.6875784190715182	(0.6650485436893203.

Rysunek 3.8: Fragment pliku w formacie *.csv* zawierającego rezultaty przeprowadzanych eksperymentów.

W czasie pracy wykorzystana została pięciokrotnie powtórzona metoda dwukrotnej walidacji krzyżowej. W ramach wszystkich zbiorów danych przeprowadzony został eksperyment badający skuteczność klasyfikacji bez wykonywania wcześniejszej selekcji cech oraz biorący pod uwagę różną liczbę cech w procesie klasyfikacji.

Liczba cech użyta do eksperymentów została wybrana na podstawie różnych kryteriów - w zależności od rodzaju eksperymentu. W większości przypadków posłużono się testem Wilcoxona w taki sposób, aby klasyfikacja zbiorów bez użycia selekcji cech i po jej użyciu dawała rezultaty bez zmian statysycznie istotnych.

Łącznie przeprowadzono około 6900 eksperymentów, po $115 \cdot \textit{liczba cech zbioru}$ dla każdej opisanej metody selekcji cech. Eksperymenty zawierają każdą permutację z zadanych zagadnień:

- porównanie wyników dla każdego z 115 zbiorów danych,
- porównanie wyników dla każdej z 5 metod selekcji,
- porównanie wyników dla danych nie poddanych selekcji cech,
- porównanie wyników każdej ilości liczby cech w ramach każdej metody,
- porównanie wydajności poszczególnych algorytmów.

Pliki źródłowe zawierające kod biblioteki, która została napisana do wygenerowania wyników znajduje się w załączonym do pracy katalogu *src*. Bazy danych, na których przeprowadzane były doświadczenia zostały również zamieszczone i znajdują się w folderze *data*. Rezultaty eksperymentów opisane zostały w rozdziale 4 - Wyniki.

4 Wyniki

Badania zostały przeprowadzone zgodnie z zaplanowanym schematem. Dla każdej testowanej metody obliczono dokładność zbalansowaną, precyzję, czułość oraz wynik testu *F1 score*. Dane te policzono **przyjmując jako elementy pozytywne macierzy konfuzji instancje klas mniej licznych**. Dalsza część rozdziału zawiera wykresy oraz tabele porównujące testowane metody.

4.1 Badanie optymalnej liczby cech i wyników klasyfikacji dla poszczególnych metod

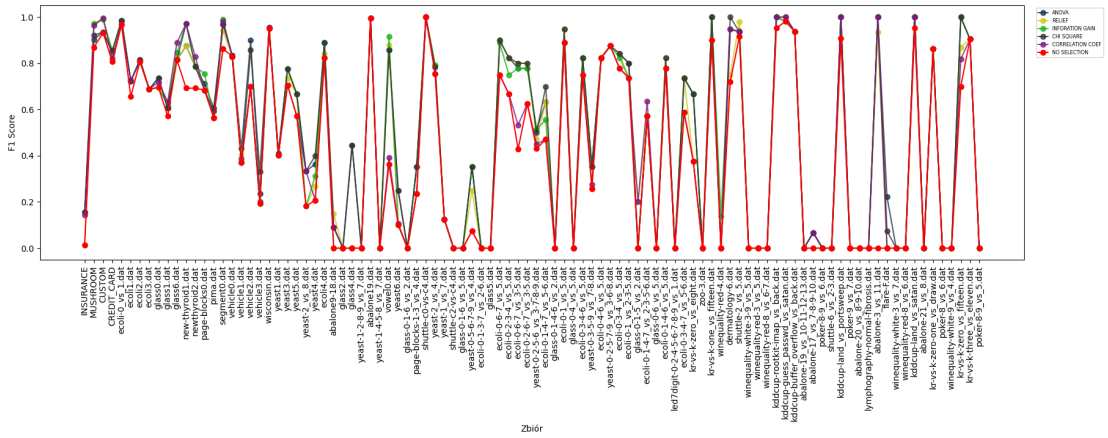
W celu przeprowadzenia klasyfikacji, pierwszym krokiem jaki należało podjąć w ramach każdej analizy było określeniej właściwej liczby cech dla badanych algorytmów i zbiorów. Zdecydowano się na przeprowadzenie badań w trzech kierunkach:

- zbadanie każdej kombinacji liczby cech dla każdej metody,
- zbadanie, dla jakiej liczby cech metoda jest w stanie uzyskać wynik klasyfikacji podobny do wyniku uzyskanego przeprowadzając klasyfikację na pełnym zbiorze,
- zbadanie, jak poszczególne metody radzą sobie w ramach takiej samej, uprzednio wybranej liczby cech.

Szczegóły dotyczące doświadczeń oraz rezultaty zostały opisane w rozdziałach 4.1.1 - 4.1.3. Pełny zestaw wygenerowanych wyników został załączony do pracy na płycie CD i znajduje się w katalogu *results*. Foldery *f1_best*, *f1_wilcoxon* oraz *percent* reprezentują kolejne trzy przedstawione powyżej kierunki badań. Katalog *classic* zawiera wyniki klasyfikacji przeprowadzonej dla trzech i pięciu naistotniejszych cech według każej metody selekcji - średniej liczby cech po podwyższeniu której obserwowano brak poprawy lub pogorszenie jakości klasyfikacji. W katalogu *f1_closest* znajduje wariacja drugiego, opisanego powyżej doświadczenia, w którym zamiast wyniku testu Wilconxona, jako metrykę użyto *F1 Score*. Folder *time* prezentuje wyniki pomiarów czasu dla poszczególnych eksperymentów. Pliki dostarczone zostały w formacie *csv*. Pierwszy rząd zawiera opis poszczególnych kolumn, pozostałe - wyniki badań.

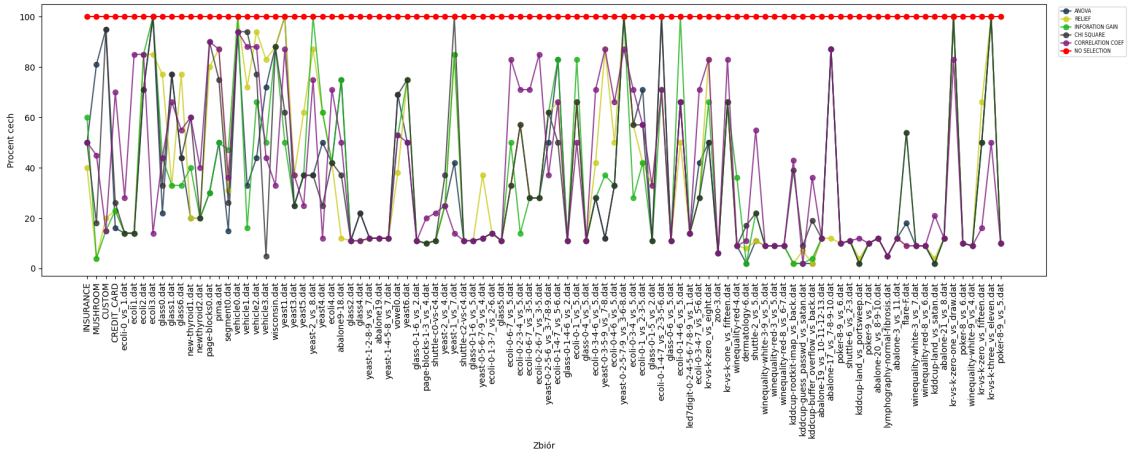
4.1.1 Kombinacja wszystkich metod i każdej możliwej liczby cech

Celem tego badania było sprawdzenie, jak każda z badanych metod może poprawić wyniki klasyfikacji. W ramach tego etapu przygotowano program, który dla każdego zbioru i dla każdej metody porównuje wyniki klasyfikacji każdej możliwej liczby cech. Po porównaniu wszystkich możliwości, algorytm zapisuje doświadczenie z najlepszym wynikiem. Kryterium obranym przy porównywaniu wyników było *F1 Score*. Wyniki, jakie udało się uzyskać dla każdego zbioru przedstawione zostały na rysunku 4.1. Jak można zaobserwować - selekcja cech poprawiła wyniki klasyfikacji w przypadku praktycznie każdego zbioru.



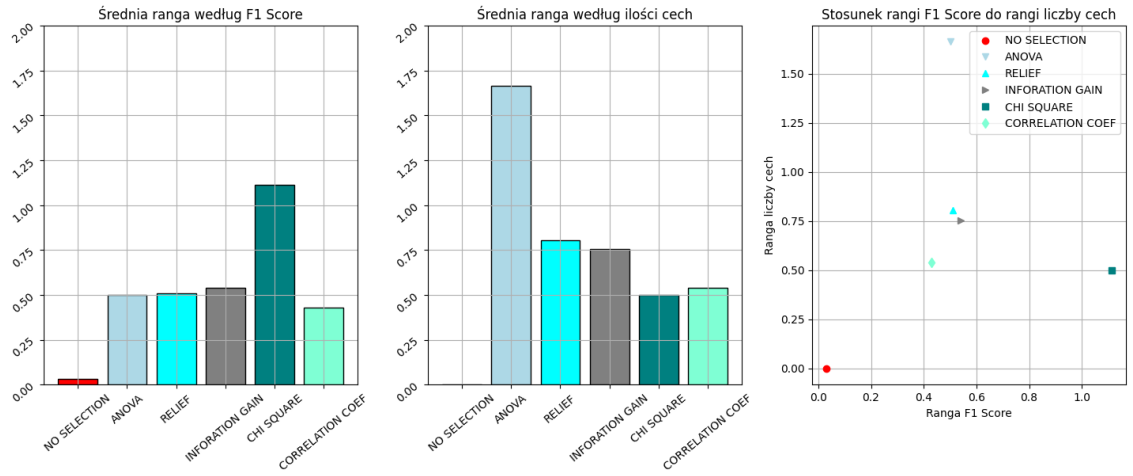
Rysunek 4.1: Wyniki F1 Score, jakie udało się uzyskać dla poszczególnych metod i zbiorów.

Wykres 4.2 prezentuje procentową liczbę cech w stosunku do liczby wszystkich cech, jaką posłużyła się dana metoda w celu wyodrębnienia najbardziej optymalnych według niej atrybutów. Rysunek zawiera porównanie dla wszystkich badanych zbiorów.



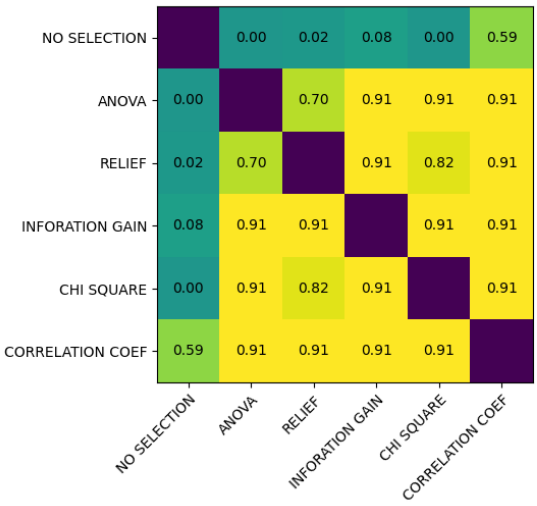
Rysunek 4.2: Procentowa liczba cech, dla której każda metoda uzyskała najlepszy wynik.

Ponieważ w przypadku tego eksperymentu każdej metodzie selekcji pozwolono wybrać najoptymalniejszą według niej liczbę cech i najlepsze według niej cechy, a także wszystkie metody trenowano na tych samych zbiorach, można pokusić się o porównanie ich uśrednionych wyników. Wykres 4.3 przedstawia uśrednione rangi dla wyników *F1 Score* oraz liczby cech, dla każdej metody i dla wszystkich 115 zbiorów. Rangi zostały przydzielone na podstawie każdego doświadczenia, w zależności od skuteczności metody. Najlepsza metoda została oznaczona rangą '5,' metoda najgorsza - '0.' W przypadku badania liczby cech, za najlepszy wynik uznano ten z najmniejszą ich ilością. W przypadku, gdy dwie metody miały ten sam wynik, otrzymywały tę samą rangę. Kolorem czerwonym oznaczono rezultat uzyskany dla klasyfikacji bez użycia żadnych metod redukcji atrybutów.



Rysunek 4.3: Uśrednione rangi F1 Score i liczby cech dla wszystkich zbiorów, dla poszczególnych metod.

Aby móc określić, ktora z badanych metod radzi sobie najlepiej, koniecznym krokiem było porównanie wyników z pomocą testu statystycznego, w celu określenia, czy pomiędzy rezultatami eksperymentów dla różnych metod występują istotne różnice statystyczne. Testem statystycznym użytym w badaniu był test Wilcoxona. Macierz prezentująca wyniki tego testu została przedstawiona na rysunku 4.4. Metryką użytą do sprawdzenia różnic statystycznych zostało *F1 Score*, natomiast metryką objaśniającą różnice statystyczne zostało *pvalue*. Aby uniknąć problemu porównań wielokrotnych wynikającego z jednoczesnego wykonywania wielu porównań z tej samej grupy hipotez zdecydowano się na skorzystanie z *metody Hommmela*, która przeciwdziała temu problemowi zmniejszając nominalny poziom dokładności każdego z testowanych zbiorów [59]. Metoda ta oparta jest na wskaźniku *FWER* (family-wise error rate) i jest szeroko stosowana, zwłaszcza gdy tylko kilka hipotez jest testowanych jednocześnie.



Rysunek 4.4: Macierz prezentująca wyniki testu Wilcoxona objaśniające różnice statystyczne pomiędzy najlepszymi wynikami wszystkich metod.

Rysunek 4.4 potwierdza to, co można było przypuszczać analizując rysunki 4.1 oraz 4.3 - wszystkie metody selekcji cech pozwalają na uzyskanie statystycznie podobnych wyników, a duże różnice istnieją jedynie pomiędzy wynikami klasyfikacji z użyciem dowolnej z metod a wynikami klasyfikacji bez redukcji cech. Wyniki testu Wilcoxona dla poszczególnych metod wskazują, że rezultaty uzyskane przez te metody nie wykazują znacznych różnic statystycznych. Żaden z wyników nie uzyskuje zwyczajowej wartości 0.05 uznawanej za próg, po którym można uznać, że rezultaty eksperymentów pochodzą z różnych rozkładów. Z tego powodu, nie można z całą pewnością stwierdzić, która metoda daje najlepsze wyniki.

Odchylenie ćwiartkowe należy do pozycyjnych miar zmienności wyników. Definiuje się je jako połowę różnicy między kwartylem trzecim Q_3 i pierwszym Q_1. Interpretuje się je jako przeciętne zróżnicowanie badanych jednostek wokół mediany [60].

Tablica 4.1 zawiera porównanie średniej liczby cech w procentach, średni wyniki *F1 Score* dla każdej metody oraz odchylenie ćwiartkowe wyników *F1 Score*, dla części badanych zbiorów (stopień niezbalansowania pomiędzy 1:1.5 a 1:9). Czcionką pogrubioną zaznaczono na tablicy wartości skrajne liczby cech.

Tablica 4.1: Porównanie średniej liczby wybranych cech, uśrednionego wyniku F1 Score oraz ochylenia ćwiarkowego wyników dla każdej z metod, dla części zbiorów (stopień niezbalansowania pomiędzy 1:1.5 a 1:9).

Nazwa Metody	Średnia liczba cech	Uśredniony wynik F1 Score	Odchylenie ćwiartkowe wyników
Anova	49.3%	0.739	0.20758
ReliefF	63.3%	0.705	0.16780

Nazwa Metody	Średnia liczba cech	Uśredniony wynik F1 Score	Odchylenie ćwiartkowe wyników
Information Gain	**47.8%**	0.711	0.19409
Chi Square	56.6%	0.732	0.20135
Correlation Coefficient	**67.5%**	0.680	0.26457

Badanie powtórzono dla wyników uzyskanych na czterech, największych zbiorach, opisanych w rozdziałach 3.4.1 - 3.4.4. Zbiory te posiadały znacznie większą liczba atrybutów, a większy ich wymiar pozwala przypuszczać, że metody mogły lepiej dostosować się do danych. Wyniki tego badania prezentuje tablica 4.2.

Tablica 4.2: Porównanie średniej liczby wybranych cech, uśrednionego wyniku F1 Score oraz odchylenia ćwiartkowego wyników dla każdej z metod, najobszerniejsze zbiory danych.

Nazwa Metody	Średnia liczba cech	Uśredniony wynik F1 Score	Odchylenie ćwiartkowe wyników
Anova	**60.8%**	0.709	0.41904
ReliefF	**21.9%**	0.705	0.55568
Information Gain	25.7%	0.742	0.47614
Chi Square	47.4%	0.716	0.42321
Correlation Coefficient	45.1%	0.730	0.49723

Z danych, przedstawionych w tablicach 4.1 oraz 4.2 wynika, że w celu uzyskania podobnych wyników etykietowania metody potrzebowały średnio różnej liczby cech. Różnice te sięgają nawet czterdziestu punktów procentowych. Może to oznaczać, że nie jakość klasyfikacji a liczba cech potrzeba do osiągniecia jej zadowalającego poziomu powinna być wyznacznikiem przy doborze odpowiedniej metody selekcji.

Tablica 4.3 prezentuje podobne badanie, tym razem z użyciem wszystkich zbiorów danch. Przy zastosowaniu większej, reprezentatywnej liczby zbiorów, dane dotyczące liczby cech dla różnych metod nie odbiegają od siebie. Oznacza to, że i pod tym względem nie da się wskazać metody najlepszej - co potwierdza zakładaną początkowo hipotezę.

Tablica 4.3: Porównanie średniej liczby wybranych cech, uśrednionego wyniku F1 Score oraz odchylenia ćwiartkowego wyników dla każdej z metod, wszystkie zbiory danych.

Nazwa Metody	Średnia liczba cech	Uśredniony wynik F1 Score	Odchylenie ćwiartkowe wyników
Anova	34.3%	0.497	0.86015
ReliefF	36.4%	0.472	0.84210
Information Gain	34.4%	0.481	0.85889
Chi Square	36.3%	0.498	0.85783
Correlation Coefficient	39.5%	0.432	0.81990

Ponadto okazało się, że w 93% przypadków (459 / 490 wyników), selekcja cech pozwoliła uzyskać lepsze wyniki klasyfikacji. W 71% przypadków (351 / 490 wyników), zaledwie połowa oryginalnych cech pozwoliła uzyskać wynik klasyfikacji lepszy od tego, który uzyskał algorytm na pełnym zbiorze. Co zaskakujące, w przypadku aż 218 eksperymentów - czyli dla 44% wyników, rezultat lepszy od bazowego uzyskano wykorzystując zaledwie jedną cechę.

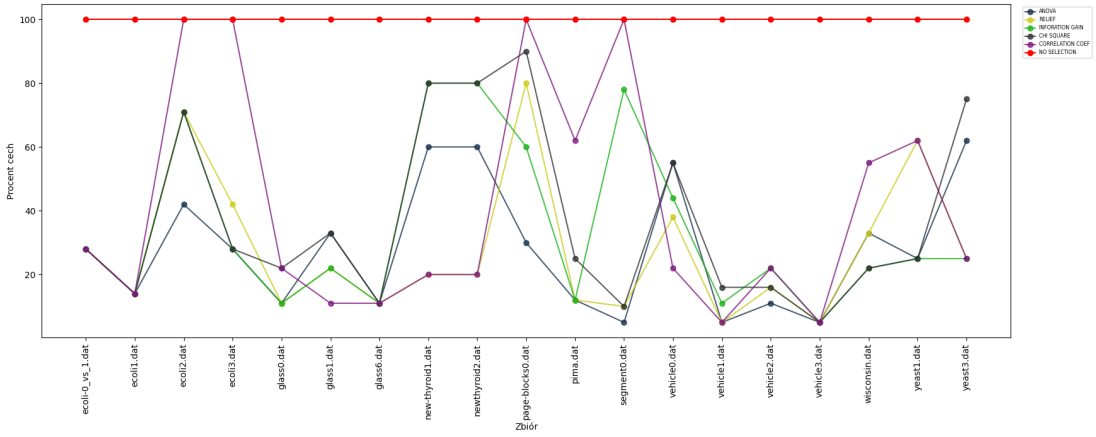
4.1.2 Badanie liczby cech dla której wyniki odpowiadają wynikom klasyfikacji na pełnym zbiorze

Celem tego badania było sprawdzenie, jaka liczba cech jest potrzebna poszczególnym metodom, aby wyniki klasyfikacji odpowiadały rezultatom badań bez przeprowadzonej redukcji atrybutów. Ma to znaczenie szczególnie wtedy, gdy od selekcji cech oczekuje się przede wszystkim zmniejszenia wymiaru wyjściowego zbioru w celu obniżenia kosztów jego archiwizacji i przetwarzania. Metryką na podstawie której badano, czy istnieje pomiędzy takimi wynikami statystycznie istotna zmiana było *F1 Score*. Badanie polegało na stworzeniu ośmiu różnych podzbiorów każdego oryginalnego zbioru i przeprowadzeniu selekcji cech oraz klasyfikacji na każdym z nich. Korzystając z wyników, zbadano z pomocą testu Wilcoxona różnice statystyczne pomiędzy rezutlatami klasyfikacji bez selekcji cech i dla każdej z metod. Przyjęto standardowe $pvalue$ wynoszące 0.05 jako wystarczające, by uznać takie wyniki za podobne - szczególnie, że test przeprowadzany był z założeniem, że elementy pozytywne macierzy konfuzji to instancje klas mniej licznych, co powoduje, że nawet jedna instancja sklasyfikowana błędnie może znacząco wpłynąć na wynik testu. Program napisany w ramach tego etapu bada wszystkie permutacje liczby cech, począwszy od jednej cechy i zapisuje rezultat dla minimalnej liczby cech, dla której klasyfikacja osiąga rezultat porównywalny z założonym celem. Algorytm programu wygląda następująco:

1. Generowanie 8 pozdbiorów oryginalnego zbioru. Każdy zawiera wszystkie cechy oryginału ale różną się elementami.

2. Przeprowadzenie klasyfikacji bez redukcji atrybutów, osobno dla każdego zbioru, zapisanie każdego wyniku w tablicy.
3. Stworzenie pętli iterującej od $i = 1$ do X, gdzie X to liczba cech zbioru, kolejno dla każdej metody.
4. Przeprowadzenie selekcji cech dla każdego podzbioru i liczby cech ustawionej na i oraz wygenerowanie wyników klasyfikacji.
5. Przeprowadzenie testu Wilconxona pomiędzy wygenerowanymi wynikami a wynikami zapisanymi w tablicy.
6. Jeżeli warunek $pvalue > 0.05$ jest spełniony - przerwanie pętli i zapisanie wyniku, w przeciwnym wypadku kontynuacja obliczeń.

Doświadczenie pozwoliło sprawdzić, jaka liczba cech jest wymagana, aby pomiędzy zbiorem stworzonym przez redukcję atrybutów a zbiorem oryginalnym nie było istotnych różnic statystycznych. Drugim celem eksperymentu było badanie stabilności metod. Rysunek 4.5 prezentuje, jaki procent cech był wymagany, aby wynik klasyfikacji zbioru przed selekcją nie odbiegał statystycznie od tego po selekcji. Wykres prezentuje wyniki dla części zbiorów danych (stopień niezbalansowania pomiędzy 1:1.5 a 1:9).



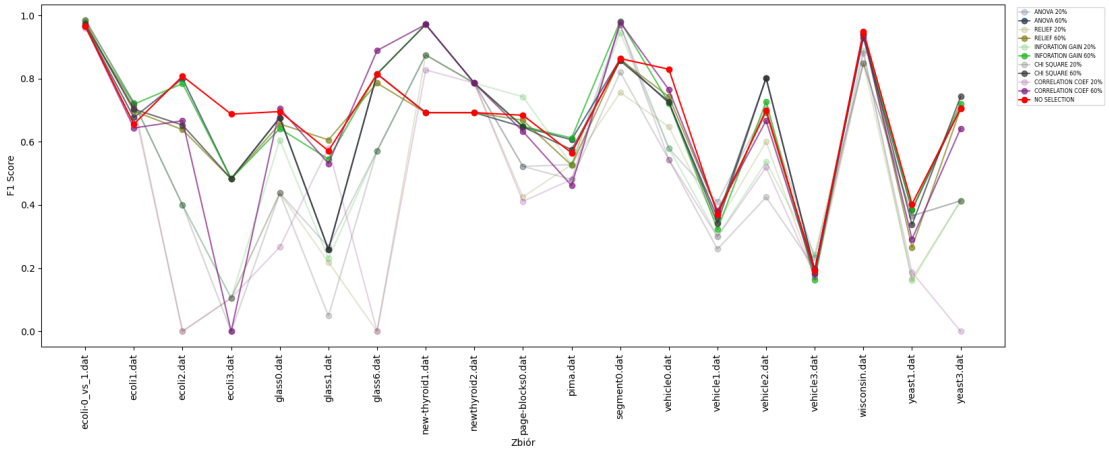
Rysunek 4.5: Procentowa liczba cech wymagana do spełniania hipotezy zerowej testu Wilcoxona dla części zbiorów danych (stopień niezbalansowania pomiędzy 1:1.5 a 1:9).

Z badania wynika, że zastosowanie selekcji cech pozwala uzyskać istotnie mniejszy wymiar zbioru przy jednoczesnym zachowaniu podobnej jakości klasyfikacji. Nie istnieje jednak jedna optymalna metoda. W ramach jednego zbioru należy przetestować kilka różnych algorytmów selekcji cech i wybrać ten, który najlepiej interpretuje dane.

4.1.3 Porównanie metod w ramach ustalonej liczby cech

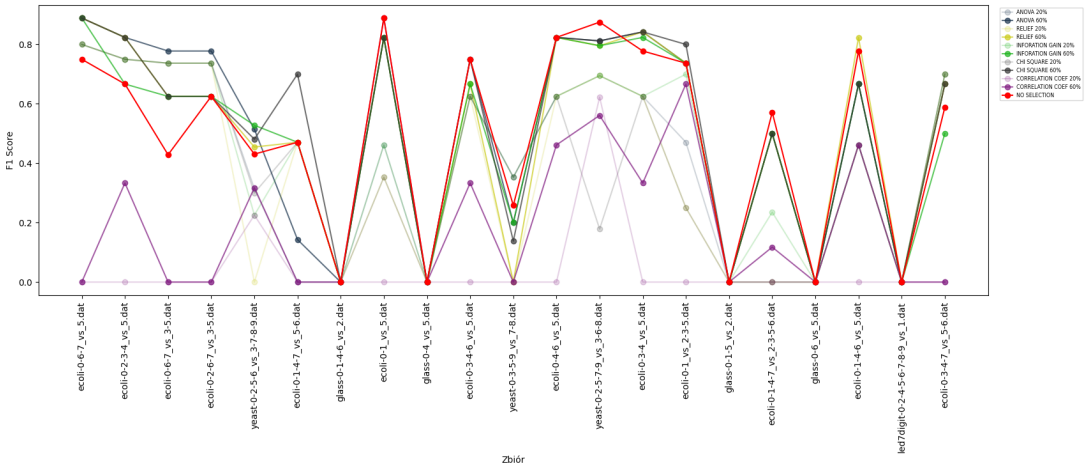
Dla obiektywnego porównania algorytmów, zdecydowano się na wybór pięciu wartości liczby cech w dla każdej metody i każdego zbioru. Program napisany w ramach analizy przeprowadza

test dla 20%, 40%, 60%, 80% i 100% liczby cech. Pozwoliło to na sprawdzenie, jak poszczególne algorytmy radzą sobie mając do dyspozycji stałą, określoną liczbę cech. Badanie pokazało, że nie zawsze większa liczba cech daje lepsze rezultaty, a podstawowy na to wpływ ma sam zbiór danych. Wyniki *F1 Score* dla poszczególnych metod (rezultaty dla 20% i 60% cech) i części zbiorów danych (stopień niezbalansowania pomiędzy 1:1.5 a 1:9) przedstawione zostały na rysunku 4.6. W ramach eksperymentu wykonano również próbę klasyfikacji dla danych niepodlegających wcześniej selekcji cech. Linią czerwoną na rysunku oznaczono wyniki testu *F1* dla klasyfikacji bez uprzedniej selekcji cech.



Rysunek 4.6: Porównanie wyników F1 Score dla części zbiorów danych (stopień niezbalansowania pomiędzy 1:1.5 a 1:9) - 20% i 60% cech.

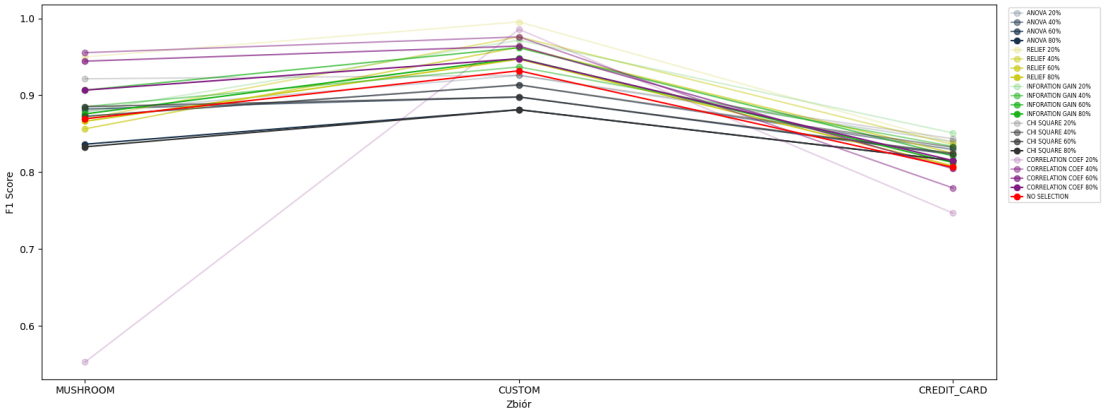
Rezultaty podobnego badania, dla zbiorów o stopniu niezbalasowania wynoszącym od 1:20 do 1:100 przedstawiono na rysunku 4.7.



Rysunek 4.7: Porównanie wyników F1 Score dla części zbiorów danych (stopień niezbalansowania pomiędzy 1:20 a 1:100) - 20% i 60% cech.

Jak widać, w przypadku tak mocno niezbalansowanych zbiorów, mała liczba cech często nie pozwala na poprawną klasyfikację danych mniejszościowych - co tłumaczy duża ilość wyników równych zero, szczególnie dla przypadku dwudziestoprocentowego. Wyraźnie słabiej radzi sobie w tym przypadku również algorytm *Correlation Coefficient*, czego nie zaobserowowano na poprzednim wykresie.

Ponownie, badania powtórzono dla największych zbiorów danych - *CCFD*, *Mushroom* oraz *Custom*. Ponieważ posiadają one znaczną liczba atrybutów oraz znacznie większy wymiar, istnieje możliwość, że metody selekcji cech lepiej dostosują się do danych. Rysunek 4.8 zawiera porównanie wyników testu F1 Score dla tych zbiorów, z uwzględnieniem pozostałych eksperymentów (40% i 80% cech).

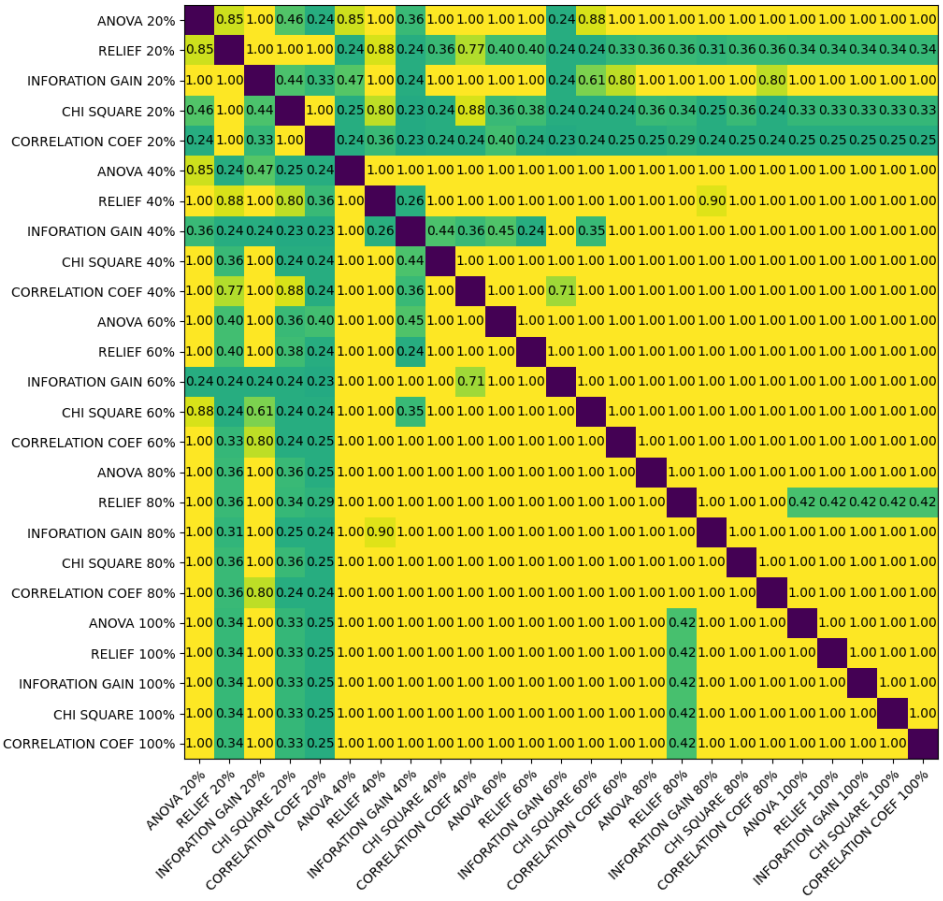


Rysunek 4.8: Porównanie wyników F1 Score dla zbiorów *CCFD*, *Mushroom* oraz *Custom* - 20%, 40%, 60% i 80% cech.

Selekcja cech praktycznie w każdym przypadku pozwoliła uzyskać lepsze rezultaty klasyfikacji. Ponownie wyróżniającą się metodą jest *Correlation Coefficient*. Słabe wyniki uzyskuje on jednak tylko w przypadku niewielkiej liczby cech, w zestawie *Custom* natomiast, radzi sobie najlepiej, co jest prawdopodobnie podyktowane silnie skorelowanymi instancjami klas mniejszościowych w tym zbiorze.

Aby potwierdzić lub obalić hipotezę postawioną na początku pracy, postanowiono przeprowadzić test statystyczny - test Wilconxona, dla wygenerowanych danych. Rysunek 4.9 prezentuje macierz prezentującą te wyniki. Pokazuje ona prawdopodobieństwo, że rezultaty uzyskane dla każdej pary eksperymentów pochodzą z tej samej dystrybucji. Poszczególne komórki zawierają wynik *pvalue*, wartość mniejsza niż 0.05 wskazuje na statystyczne różnice otrzymanych rezultatów tym samym odrzucając hipotezę. Podobnie jak w doświadczeniu opisanym w rodziale 4.1.1, podczas wykonywania testów skorzystano z poprawki przeciwdziałającej problemowi porównań wielokrotnych. Uważa się jednak, że *FWER* jest zbyt konserwatywny w przypadkach, gdy liczba jednocześnie testowanych hipotez sięga już kilkuset [59]. Z uwagi na dużą ilość porównywanych zbiorów zdecydowano się na *metodę Benjamini i Yekutieli*, bazującą na innym wskaźniku - *FDR* (false discovery rate), który jest mniej rygorystyczny i używa się go podczas testowania dużej

liczby hipotez [59].



Rysunek 4.9: Macierz prezentująca wyniki testu Wilcoxona objaśniające różnice statystyczne pomiędzy metodami z różną liczbą cech.

Obserwując wykres, można generalizować, że różnice statystyczne pomiędzy metodami maleją wraz ze wzrostem liczby cech. Użycie już 40% cech odrzuca definitywnie hipotezę o niezależnych dystrybucjach, potwierdzając niejako, że każda z użytych metod skutkuje podobnymi wynikami. Ponadto, rezultaty przedstawione na rysunkach 4.6 - 4.8 dowodzą, że 20% to zazwyczaj liczba zbyt mała liczba cech do uzyskania stabilnych, dobrych wyników klasyfikacji. Wyniki eksperymentu świadczą za hipotezą postawioną na początku pracy.

4.2 Badanie różnic wydajnościowych

Podczas doświadczeń badano czas, jaki potrzebny jest poszczególnym eksperymentom na ukończenie obliczeń. Każdą z opisanych metod uruchomiono kilkuktrotnie na różnych zbiorach, a jako parametr określający liczba cech podano całkowitą liczbę cech w każdym zbiorze. Wyniki tych eksperymentów prezentuje tablica 4.4:

Tablica 4.4: Uśredniony czas wykonywania poszczególnych algorytmów oraz ich deklarowane złożoności czasowe.

Nazwa Metody	Średni czas wykonywania (s)	Średni czas / (liczba cech * liczba elementów)	Złożoność czasowa
Anova	0.00086405	0.0000002	$O(n)$
ReliefF	2.62687309	0.0001468	$O(m^2 * n)$
Information Gain	0.07214137	0.0000094	$O(m * n)$
Chi Square	0.00167353	0.0000004	$O(n)$
Correlation Coefficient	0.00315445	0.0000006	$O(n)$

Pod względem wydajności metody nie różnią się ze sobą znacząco - z dwoma wyjątkami. Metoda *ReliefF* okazała się kilkusetkrotnie wolniejsza od konkurentów. Ma to swoje uzasadnienie w algorytmie tej metody - najbardziej obciążającym elementem programu jest konieczność stworzenia macierzy odległości typu *manhattan* dla każdej cechy i każdej instancji, a następnie przeiterowania się po każdym elemencie tej macierzy. Zgadza się to z wyliczeniami twórców tego algorytmu - według nich *ReliefF*, posiada złożoność $O(m^2 * n)$, gdzie m jest liczbą instancji, natomiast n - liczbą cech [61]. Drugim wyjątkiem jest metoda *Information Gain*, która przez konieczność "maksymalizowania zysku" osiąga złożoność $O(m * n)$, gdzie m to liczba instancji, natomiast n - liczba cech [62] [63]. Pozostałe algorytmy posiadają deklarowaną złożoność obliczeniową rzędu $O(n)$, co zgadza się z wynikami otrzymanymi w doświadczeniach [64].

Hipoteza postawiona na początku pracy nie uwzględniała różnic w czasie wykonywania algorytmu, ale jest to element na który zdecydowanie warto zwrócić uwagę decydując się na pracę z daną metodą.

5 Wnioski

Przeprowadzone badania częściowo potwierdzają założoną hipotezę. Przy odpoowiednio dobranej liczbie cech wszystkie metody poradziły sobie dobrze z powierzonym zadaniem, a żaden algorytm pod względem jakości etykietowania nie odbiegał znacząco od pozostałych. Poprawa rezultatów klasyfikacji i możliwość zmniejszenia wymiaru zbioru bez szkody dla wyników świadczy o tym, że selekcja cech jest ważnym krokiem w procesie zadania klasyfikacji. Poniżej zaprezentowano interpretację wyników uzyskanych w rodziale 4, a także opisano praktyczne zastosowanie przeprowadzonych analiz i dalsze możliwości rozwoju projektu.

5.1 Podsumowanie

Ekpseryment 4.1.1 pokazał, że redukcja atrybutów może mieć bardzo pozytywny wpływ na wyniki klasyfikacji danych. Wśród rezultatów, w przypadku niektórych zbiorów zaobserwować można nawet stuprocentową poprawę jakości klasyfikacji dla danych mniejszościowych, a sama selekcja cech pozwoliła poprawić wyniki klasyfikacji w 95% przypadków, co potwierdzone zostało testami statystycznymi. Te same testy wykazały również, że wyniki poszczególnych metod pochodzą ze statystycznie podobnych rozkładów co potwierdza hipotezę postawioną przez autora. Warto jednak zwrócić uwagę, że doświadczenie opisane w tym rodziale ukazuje tak naprawdę najlepsze wyniki uzyskane przez każdą z metod dla każdego zbioru, a wyniki opisane w rodziałach 4.1.2 oraz w szczególności w 4.1.3 obrazują, że nawet niewielka zmiana w zbiorze może zmienić działanie algorytmu redukcji cech. W dodatku, dane w tablicach 4.1 i 4.2 dowodzą, że różne algorytmy, w ramach jednego zbioru, potrzebują czasem bardzo różnej liczby cech do osiągnięcia zadowalających wyników. Poprzestawanie na samej selekcji cech nie przyniesie tak dobrych rezulatatów jakościowych jak połączenie jej z innymi metodami, co - jak zostało to opisane w przeglądzie literatury - jest bardzo popularną praktyką wśród autorów podobnych prac.

Nie mniej istotnym aspektem jest liczba cech, jaka konieczna była do uzyskania takich wyników. W 44% przypadków, eksperyment wykazał wynik klasyfikacji taki sam lub lepszy już po użyciu jednej cechy. Pokazuje to, że istnieje możliwość zmniejszenia wymiarowości istniejących baz danych, co przekłada się bezpośrednio na krótszy czas klasyfikacji oraz mniejszą ilość miejsca zajmowanego na dyskach. Jest to jeden z najważniejszych wniosków płynących z tej pracy. Prezentuje to w szczególności własnie badanie opisane w akapicie 4.1.2. Selekcja cech jest bardzo dobrym sposobem redukcji wymiarowości bez szkody dla istotnych danych. Doświadczenia przeprowadzone w ramach tego eksperymentu obrazują, że z użyciem ułamka początkowej liczby cech jest możliwe wygenerowanie znacznie mniejszego zbioru, nieróżniącego się statystycznie od pierwowzoru. W przypadku danych niezbalansowanych to nie poprawa jakości a zmniejszenie wymiaru jest prawdziwą wartością dodaną, pozwalającą na jednoczesne zarządzanie znacznie większymi bazami danych oraz na łatwiejsze ich magazynowanie.

Doświadczenia opisane w rozdziale 4.1.3 również dostyczą hipotezy postawionej na początku pracy. Eksperymenty te bezpośrednio porównują każdą z badanych metod, uruchomioną dla takiej samej liczby cech w ramach każdego zbioru. Algorytmy radziły sobie bardzo różnie, nie zawsze wynikiem ich działania było podniesienie jakości klasyfikacji. Nie da się też w łatwy sposób uogólnić działania poszczególnych metod, a nawet poszczególnych metod z poszczególną liczbą cech, co pokazuje macierz przedstawiona na rysunku 4.9. Wśród przedstawionych wyników znaleźć można przykłady, w których mała liczba cech danej metody obniżyła wynik *F1 Score* badania do poziomu najniższego ze wszystkich ekpserymentów, aby przy następnym zbiorze, doświadczenie z tymi samymi parametrami zaowocowało bardzo dobrymi rezultatami. Ważnym wnioskiem, wynikającym z tego doświaczenia jest opisany już przez autorów innych, wymienionych w przeglądzie literatury fakt, że wśród popularnych metod - nie tylko selekcji cech - nie da się w łatwy sposób wybrać lepszych oraz gorszych algorytmów. Do każdego zbioru danych należy podejść indywidualnie. To w sporej części wiedza dziedzinowa dotycząca wartości w zbiorze, pozwoli na wybór najlepszych narzędzi do pracy, co za tym idzie - uzyskanie najlepszych wyników. Badania skupiające się głównie nad zbiorami niezbalansowanymi jeszcze bardziej podkreślają ten fakt.

Warto mieć na uwadze, że selekcja cech nie zawsze ma pozytywny wpływ na jakość klasyfikacji, gdyż cechy czasem pozornie tylko nieistotne mogą mieć znaczenie dla poprawnego sklasyfikowania elementów. Pokazują to dobrze przykłady zobrazowane na rysunku 4.7. Niektóre algorytmy, mimo że zauważająco polepszyły wydajność kategoryzacji, to przyczyniły się też do spadku dokładności. Ważne jest odpowiednie dobranie metody i liczby cech - odpowiednio dobrany do problemu algorytm redukcji nie tylko nie wpłynie negatywnie na jakość klasyfikacji, ale może również - szczególnie w przypadku zbiorów zawierających dużą liczba elementów i klas - wpływać pozytywnie na czas działania algorytmu.

Analiza wszystkich doświadczeń uwidacznia jeszcze jeden fakt, zaprezentowany na przykład na rysunkach 4.1, 4.7 i 4.8. Selekcja cech dobrze sobie radzi w przypadku dużych zbiorów danych, nawet jeżeli posiadają bardzo wysoki stopień niezbalansowania (dla przykładu, w zbiorze *CCFD*, klasa mniejszościowa zajmowała zaledwie 0,172% całego zbioru, ale zbiór posiada 284807 elementów) oraz w przypadku małych zbiorów z niewielkim stopniem niezbalansowania. Jednakże, dla zbiorów o małym wymiarze (większość zbiorów z platformy *KEEL* zawierała poniżej 2000 elementów) i wysokim stopniu niezbalasowania (1:100 - 1:400) nie należy oczekiwać dobrych wyników klasyfikacji, jeżeli to selekcja cech była głównym punktem przygotowania zbiorów.

Jedynym kryterium, według którego możliwe jest obiektywne wskazanie faworytów jest kryterium wydajnościowe. Testowane metody odznaczają się różną złożonością co przekłada się na różny czas działania. W skrajnych przypadkach metoda *ReliefF* potrafiła być kilka tysięcy razy wolniejsza od metod *Anova*, *Information Gain* czy też *Correlation Coefficient*.

Podsumowując, podczas stosowania algorytmów uczenia maszynowego, należy pamiętać, że selekcja cech jest jedynie jednym z wielu dostępnych narzędzi i nie należy traktować jej jako złotego środka. Pozwala jednak, szczególnie dla dużych zbiorów danych, na wybór cech istotnych z punktu widzenia klasyfikacji, co może przysłużyć się nie tylko problemowi wydajności

współczesnych komputerów, ale również optymalizacji baz danych, które mogą dzięki temu przechowywać jedynie dane ważne dla rozpatrywanego problemu.

5.2 Praktyczne zastosowanie eksperymentów

Wykonana przez autora analiza pozwala świadomie stosować różne metody selekcji cech w projekcie wykorzystującym uczenie maszynowe. Badania przeprowadzone w ramach pracy umożliwiają uświadomienie sobie wagi tego etapu w procesie kasyfikacji danych szczególnie w przypadku zbiorów o dużej wymiarowości lub nierównomiernej dystrybucji klas. Zastosowanie dużej liczby zbiorów pozwala prześledzić, jak poszczególne metody radzą sobie z różnym stopniem niezbalansowania i z różnymi wymiarami bazy danych, analiza wyników daje pogląd na to, jak prawidłowo zestwiać rezultaty przeprowadzonych badań.

5.3 Dalsze możlowości rozwoju

W ramach dalszego rozwoju pracy należałoby sprawdzić inne rodzaje metod selekcji cech - metody opakowane oraz metody wbudowane. Przy odpowiednim podejściu do testowania tych algorytmów, możliwe byłoby porównanie, jak radzą sobie one na tle przedstawionych metod rankingowych pod względem wydajności oraz jakości klasyfikacji. Ciekawym aspektem wartym uwagi jest również analiza porównawcza przytoczonych metod dla zbiorów wieloklasowych. Doświaczenia zorientowane w tym kierunku pozwoliłyby stwierdzić, czy większa liczba klas ma wpływ na klasyfikację danych mniejszościowych i jak selekcja cech może pomóc taki wypływ minimalizować.

6 Bibliografia

[1] G. Weiss and F. Provost, "Learning when training data are costly: The effect of class distribution on tree induction," *J. Artif. Intell. Res. (JAIR)*, vol. 19, pp. 315–354, Jul. 2003, doi: 10.1613/jair.1199.

[2] M. Hall, "Correlation-based feature selection for machine learning," *Department of Computer Science*, vol. 19, Jun. 2000.

[3] G. Abdulrauf Sharifai and Z. Zainol, "Feature selection for high-dimensional and imbalanced biomedical data based on robust correlation based redundancy and binary grasshopper optimization algorithm," *Genes*, vol. 11, no. 7, 2020, doi: 10.3390/genes11070717.

[4] L. Yin, Y. Ge, K. Xiao, X. Wang, and X. Quan, "Feature selection for high-dimensional imbalanced data," *Neurocomputing*, vol. 105, pp. 3–11, 2013, doi: https://doi.org/10.1016/j.neucom.2012.04.039.

[5] B. Xue, M. Zhang, W. N. Browne, and X. Yao, "A survey on evolutionary computation approaches to feature selection," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 4, pp. 606–626, 2016, doi: 10.1109/TEVC.2015.2504420.

[6] D. Tiwari, "Handling class imbalance problem using feature selection," Apr. 2014.

[7] M. Wasikowski and X. Chen, "Combating the small sample class imbalance problem using feature selection," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, pp. 1388–1400, 2010.

[8] C. Elkan, "Magical thinking in data mining: Lessons from CoIL challenge 2000," *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Jul. 2001, doi: 10.1145/502512.502576.

[9] I. Guyon and A. Elisseeff, "An introduction of variable and feature selection," *J. Machine Learning Research Special Issue on Variable and Feature Selection*, vol. 3, pp. 1157–1182, Jan. 2003, doi: 10.1162/153244303322753616.

[10] P. Ziemba, "Redukcja wymiarowości i selekcja cech w zadaniach klasyfikacji i regresji z wykorzystaniem uczenia maszynowego," *Zeszyty Naukowe Uniwersytetu Szczecińskiego. Studia Informatica*, pp. 221–236, 2012.

[11] G. Chandrashekar and F. Sahin, "A survey on feature selection methods," *Comput. Electr. Eng.*, vol. 40, no. 1, pp. 16–28, Jan. 2014, doi: 10.1016/j.compeleceng.2013.11.024.

[12] S. Maldonado and R. Weber, "A wrapper method for feature selection using support vector machines," *Information Sciences*, vol. 179, no. 13, pp. 2208–2217, 2009, doi: https://doi.org/10.1016/j.ins.2009.02.014.

[13] J. M. Pereira, M. Basto, and A. F. da Silva, “The logistic lasso and ridge regression in predicting corporate failure,” *Procedia Economics and Finance*, vol. 39, pp. 634–641, 2016, doi: https://doi.org/10.1016/S2212-5671(16)30310-0.

[14] L. E. Melkumova and S. Ya. Shatskikh, “Comparing ridge and LASSO estimators for data analysis,” *Procedia Engineering*, vol. 201, pp. 746–755, 2017, doi: https://doi.org/10.1016/j.proeng.2017.09.615.

[15] H. Motoda and H. Liu, “Feature selection, extraction and construction,” *Communication of IICM (Institute of Information and Computing Machinery, Taiwan)*, vol. 5, pp. 67–72, Jan. 2002.

[16] A. Sophian, G. Y. Tian, D. Taylor, and J. Rudlin, “A feature extraction technique based on principal component analysis for pulsed eddy current NDT,” *NDT & E International*, vol. 36, no. 1, pp. 37–41, 2003, doi: https://doi.org/10.1016/S0963-8695(02)00069-5.

[17] L. Mena and J. Gonzalez, “Machine learning for imbalanced datasets: Application in medical diagnostic.” in *FLAIRS 2006 - Proceedings of the Nineteenth International Florida Artificial Intelligence Research Society Conference*, Jan. 2006, vol. 2006, pp. 574–579.

[18] D. A. N. S. Silva, L. C. Souza, and G. H. M. B. Motta, “An instance selection method for large datasets based on markov geometric diffusion,” *Data & Knowledge Engineering*, vol. 101, pp. 24–41, 2016, doi: https://doi.org/10.1016/j.datak.2015.11.002.

[19] H. Han, W.-Y. Wang, and B.-H. Mao, “Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning,” in *Advances in intelligent computing*, 2005, pp. 878–887.

[20] N. Chawla, K. Bowyer, L. Hall, and W. Kegelmeyer, “SMOTE: Synthetic minority over-sampling technique,” *J. Artif. Intell. Res. (JAIR)*, vol. 16, pp. 321–357, Jun. 2002, doi: 10.1613/jair.953.

[21] S.-J. Yen and Y.-S. Lee, “Cluster-based under-sampling approaches for imbalanced data distributions,” *Expert Systems with Applications*, vol. 36, pp. 5718–5727, Jan. 2006, doi: 10.1016/j.eswa.2008.06.108.

[22] H. He, Y. Bai, E. Garcia, and S. Li, “ADASYN: Adaptive synthetic sampling approach for imbalanced learning,” in *Proceedings of the International Joint Conference on Neural Networks*, Jul. 2008, pp. 1322–1328, doi: 10.1109/IJCNN.2008.4633969.

[23] J. L. Leevy, T. M. Khoshgoftaar, R. A. Bauder, and N. Seliya, “A survey on addressing high-class imbalance in big data,” *Journal of Big Data*, vol. 5, no. 1, p. 42, Nov. 2018, doi: 10.1186/s40537-018-0151-6.

[24] J. R. Quinlan, “Bagging, boosting, and C4.5,” in *In proceedings of the thirteenth national conference on artificial intelligence*, 1996, pp. 725–730.

[25] T. Chengsheng, L. Huacheng, and X. Bing, "AdaBoost typical algorithm and its application research," *MATEC Web of Conferences*, vol. 139, p. 00222, Jan. 2017, doi: 10.1051/matecconf/201713900222.

[26] K. Hempstalk, E. Frank, and I. Witten, "One-class classification by combining density and class probability estimation," Sep. 2008, doi: 10.1007/978-3-540-87479-9_51.

[27] H. J. Shin, D.-H. Eom, and S.-S. Kim, "One-class support vector machines—an application in machine fault detection and classification," *Computers & Industrial Engineering*, vol. 48, no. 2, pp. 395–408, 2005, doi: https://doi.org/10.1016/j.cie.2005.01.009.

[28] S. Ertekin, J. Huang, L. Bottou, and C. Giles, "Learning on the border: Active learning in imbalanced data classification," in *International Conference on Information and Knowledge Management, Proceedings*, Jan. 2007, pp. 127–136, doi: 10.1145/1321440.1321461.

[29] M. Langarizadeh and F. Moghbeli, "Applying naive bayesian networks to disease prediction: A systematic review," *Acta Informatica Medica*, vol. 24, p. 364, Oct. 2016, doi: 10.5455/aim.2016.24.364-369.

[30] M. Bader-El-Den, E. Teitei, and T. Perry, "Biased random forest for dealing with the class imbalance problem," *IEEE Transactions on Neural Networks and Learning Systems*, vol. PP, pp. 1–10, Nov. 2018, doi: 10.1109/TNNLS.2018.2878400.

[31] P. van der Putten and M. van Someren, "A bias-variance analysis of a real world learning problem: The CoIL challenge 2000," *Machine Learning*, vol. 57, no. 1, pp. 177–195, Oct. 2004, doi: 10.1023/B:MACH.0000035476.95130.99.

[32] D. Mladenić and M. Grobelnik, "Feature selection for unbalanced class distribution and naive bayes." Jan. 1999, pp. 258–267.

[33] G. Forman, "An extensive empirical study of feature selection metrics for text classification," *J. Mach. Learn. Res.*, vol. 3, no. null, pp. 1289–1305, Mar. 2003.

[34] Z. Zheng, X. Wu, and R. Srihari, "Feature selection for text categorization on imbalanced data," *SIGKDD Explor. Newsl.*, vol. 6, no. 1, pp. 80–89, Jun. 2004, doi: 10.1145/1007730.1007741.

[35] I. Jamali, M. Bazmara, and S. Jafari, "Feature selection in imbalance data sets," *International Journal of Computer Science Issues*, vol. 9, p. 42, May 2013.

[36] J. Biesiada and W. Duch, "Feature selection for high-dimensional data — a pearson redundancy based filter," in *Computer recognition systems 2*, 2007, pp. 242–249.

[37] I. Kononenko, E. Šimec, and M. Robnik-Sikonja, "Overcoming the myopia of inductive learning algorithms with RELIEFF," *Applied Intelligence*, vol. 7, pp. 39–55, Jan. 1997, doi: 10.1023/A:1008280620621.

[38] R. J. Urbanowicz, M. Meeker, W. La Cava, R. S. Olson, and J. H. Moore, "Relief-based feature selection: Introduction and review," *Journal of Biomedical Informatics*, vol. 85, pp. 189–203, 2018, doi: https://doi.org/10.1016/j.jbi.2018.07.014.

[39] K. Kira and L. A. Rendell, "A practical approach to feature selection," in *Machine learning proceedings 1992*, D. Sleeman and P. Edwards, Eds. San Francisco (CA): Morgan Kaufmann, 1992, pp. 249–256.

[40] M. Kumar, N. Rath, A. Swain, and S. Rath, "Feature selection and classification of microarray data using MapReduce based ANOVA and k-nearest neighbor," *Procedia Computer Science*, vol. 54, pp. 301–310, Dec. 2015, doi: 10.1016/j.procs.2015.06.035.

[41] M. G. Larson, "Analysis of variance," pp. 115–121, Jun. 2008, doi: https://doi.org/10.1161/CIRCULATIONAHA.107.654335.

[42] G. Guo, H. Wang, D. Bell, Y. Bi, and K. Greer, "KNN model-based approach in classification," in *On the move to meaningful internet systems 2003: CoopIS, DOA, and ODBASE*, 2003, pp. 986–996.

[43] F. Nahm, "Nonparametric statistical tests for the continuous data: The basic concept and the practical use," *Korean Journal of Anesthesiology*, vol. 69, p. 8, Feb. 2016, doi: 10.4097/kjae.2016.69.1.8.

[44] Y. Jung, "Multiple predicting k-fold cross-validation for model selection," *Journal of Nonparametric Statistics*, vol. 30, no. 1, pp. 197–215, 2018, doi: 10.1080/10485252.2017.1404598.

[45] "Sklearn model_selection KFold," *scikit*. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.KFold.html.

[46] M. L. G.-. ULB, "Credit card fraud detection," *Kaggle*. Mar. 2018, [Online]. Available: https://www.kaggle.com/mlg-ulb/creditcardfraud.

[47] A. Kumar, "Health insurance cross sell prediction," *Kaggle*. Sep. 2020, [Online]. Available: https://www.kaggle.com/anmolkumar/health-insurance-cross-sell-prediction.

[48] U. M. Learning, "Mushroom classification," *Kaggle*. Dec. 2016, [Online]. Available: https://www.kaggle.com/uciml/mushroom-classification.

[49] "Sklearn datasets make_classification," *scikit*. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_classification.html.

[50] "KEEL: Software tool. Evolutionary algorithms for data mining," *SCI2S*. [Online]. Available: https://sci2s.ugr.es/keel/imbalanced.php.

[51] "Sklearn preprocessing RobustScaler," *scikit*. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.RobustScaler.html.

[52] "Sklearn preprocessing LabelEncoder," *scikit*. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.LabelEncoder.html.

[53] "Sklearn feature_selection f_classif," *scikit*. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.f_classif.html.

[54] "Sklearn feature_selection chi2," *scikit*. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.chi2.html.

[55] "Sklearn feature_selection mutual_info_classif," *scikit*. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.mutual_info_classif.html.

[56] J. Li *et al.*, "Feature selection: A data perspective," *ACM Computing Surveys (CSUR)*, vol. 50, no. 6, p. 94, 2018.

[57] Jundongl, "Jundongl/scikit-feature," *GitHub - Scykit-feature*. [Online]. Available: https://github.com/jundongl/scikit-feature.

[58] "Python: Pandas DataFrame.corr()," *pandas*. Apr. 2020, [Online]. Available: https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.corr.html.

[59] S.-Y. Chen, Z. Feng, and X. Yi, "A general introduction to adjustment for multiple comparisons," *Journal of Thoracic Disease*, vol. 9, pp. 1725–1729, Jun. 2017, doi: 10.21037/jtd.2017.05.34.

[60] Cz. Domański, *Metody statystyczne: Teoria i zadania*. Wydawnictwo Uniwersytetu Łódzkiego, 2001.

[61] L. C. W. Urbanowicz RJ Meeker M, "Relief-based feature selection: Introduction and review. J biomed inform," *ACM Computing Surveys (CSUR)*, pp. 189–203, 2018.

[62] S. Raschka and V. Mirjalili, *Python Machine Learning, 3rd Ed.*, 3rd ed. Birmingham, UK: Packt Publishing, 2019.

[63] J. Su and H. Zhang, "A fast decision tree learning algorithm," in *Proceedings of the 21st national conference on artificial intelligence - volume 1*, 2006, pp. 500–505.

[64] M. Ahsan, R. Gomes, Md. M. Chowdhury, and K. E. Nygard, "Enhancing machine learning prediction in cybersecurity using dynamic feature selector," *Journal of Cybersecurity and Privacy*, vol. 1, no. 1, pp. 199–218, 2021, doi: 10.3390/jcp1010011.

Zawartość płyty CD

Do pracy dołączono płytę CD o następującej zawartości:

- kody źródłowy biblioteki, programu eksperymentalnego i programów generujących wykresy znajdujące 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.