Porównanie metod oversamplingu dla zadania klasyfikacji danych niezbalansowanych

Katarzyna Czak, Robert Walery

Politechnika Wrocławska, Wybrzeże Stanisława Wyspiańskiego 27, 50-370 Wrocław, Poland
https://pwr.edu.pl

Streszczenie Klasyfikacja danych niezbalansowanych to bardzo często występujący w uczeniu maszynowym problem. W celu jego rozwiązania używane są różne metody preprocessingu. Jednym z najpopularniejszych podejść do rozwiązania tego problemu jest oversampling, którym zajmiemy się w tej pracy. Opiszemy, zbadamy i porównamy wybrane metody oversmaplingu w celu ustalenia, która z nich jest najbardziej efektywna.

Keywords: Uczenie maszynowe · Dane niezbalansowane · Oversampling

1 Wprowadzenie

Domyślnym założeniem przy klasyfikacji danych za pomocą algorytmów uczenia maszynowego jest podobna liczba obiektów w badanych klasach. Kiedy jednak założenie to nie jest spełnione, pojawia się problem - klasyfikacja danych niezbalansowanych. Dane niezbalansowane to taki typ danych, w których liczba przykładów w jednej z klas jest znacznie mniejsza niż w pozostałych klasach. Przykładem takiego zbioru danych mogą być dane wykorzystywane do detekcji oszustw w transakcjach, gdzie liczba oszustw jest zazwyczaj znacznie mniejsza niż liczba poprawnych transakcji [2].

Klasyfikacja danych niezbalansowanych jest problemem, który może mieć poważne konsekwencje w praktyce. Niewłaściwa klasyfikacja, zwłaszcza w przypadku detekcji oszustw, może prowadzić do poważnych strat finansowych. Innym przykładem, gdzie niewłaściwa klasyfikacja może spowodować znaczące skutki, jest diagnozowanie choroby nowotworowej. Sytuacja, w której klasyfikator stwierdzi, że pacjent jest zdrowy, a w rzeczywistości jest chory niesie ze sobą dużo większe konsekwencje niż sytaucja, w której zdrowy pacjent zostanie sklasyfikowany jako chory. Dlatego też ważne jest, aby znaleźć najlepsze metody radzenia sobie z niezbalansowanymi danymi.

W tym projekcie skupimy się na porównaniu różnych metod preprocessingu, czyli operacji wykonywanych na danych przed samym procesem klasyfikacji. Preprocessing to ważna część procesu klasyfikacji, ponieważ pozwala na przygotowanie danych w taki sposób, aby klasyfikacja była bardziej dokładna i skuteczna. Popularnymi podejściami do rozwiązywania problemu danych niezbalansowanych w klasyfikacji jest oversampling i undersampling.

2 Wybrane metody oversamplingu

Oversampling polega na generowaniu przypadków klasy mniejszościowej, aby zrównoważyć liczbę przypadków w obu klasach. W ramach oversamplingu można wykorzystać różne metody, takie jak duplikacja przypadków, dodanie szumu do istniejących przypadków, wykorzystanie algorytmów generujących sztuczne przykłady np. SMOTE. Z kolei undersampling polega na zmniejszeniu liczby przypadków z klasy większościowej, aby zrównoważyć liczbę przypadków w obu klasach. Można to osiągnąć poprzez losowe usunięcie przykładów lub wykorzystanie algorytmów, które wybierają tylko pewną część przypadków z klasy przeważającej [9].

2.1 SMOTE

Jedną z najpopularnieszych metod oversamplingu jest algorytm przedstawiony w artykule [1]. Jak sama nazwa (Synthetic Minority Over-sampling Technique) wskazuje jest to technika generująca syntetyczne dane dla klasy mniejszości. W przeciwieństwie do metod undersamplingu, które usuwają przykłady większej klasy, SMOTE nie traci żadnych danych treningowych.

Algorytm działa w oparciu o analizę przestrzeni cech przykładów mniejszej klasy, aby wygenerować sztuczne przykłady. Aby to zrobić, SMOTE wybiera losowo jeden z przykładów mniejszej klasy i znajduje jego k najbliższych sąsiadów. Następnie wybierany jest jeden z tych sąsiadów i generowany jest sztuczny punkt między wybranym przykładem a wybranym sąsiadem. Algorytm powtarza ten proces określoną liczbę razy, aż rozkład danych w klasach zostanie zrównany.

W związku z tym procedura ta skupia się bardziej na przestrzeni cech, a nie na przestrzeni danych, tj. algorytm opiera się na wartościach cech i ich wzajemnych związkach, a nie na traktowaniu punktów danych jako całości.

2.2 ADASYN

ADASYN (Adaptive Synthetic Sampling) to również algorytm oversamplingu. Jak piszą autorzy artykułu [5], algorytm ten wykorzystuje podobną zasadę działania co algorytm SMOTE, ale dodaje wagę dla każdego punktu mniejszościowego, aby bardziej skupić się na generowaniu syntetycznych przykładów w trudnych do klasyfikacji obszarach przestrzeni cech.

Na początku ADASYN losowo wybiera punkt z klasy mniejszościowej i znajduje jego k najbliższych sąsiadów z innych klas. Następnie obliczana jest waga dla każdego z tych punktów na podstawie proporcji klasy w jego otoczeniu. Obszary próbek o wyższej wadze zawierają więcej przykładów klas większościowych i są trudniejsze do nauczenia. Z tego powodu w obszarach tych generowana jest większa ilość syntetycznych próbek, co czyni ten algorytm adaptacyjnym. Do syntetycznych przykładów dodawany jest biały szum, aby nowe dane były jeszcze bardziej realistyczne. Ponadto zamiast interpolacji liniowej można generować punkty na płaszczyznach wyznaczonych pomiędzy trzema punktami. ADASYN ma tendencję do zmiany granic decyzyjnych bardziej niż SMOTE. Wynika to

z faktu, że ADASYN generuje więcej syntetycznych przykładów w obszarach, gdzie granice decyzyjne są niejednoznaczne.

W artykule [5] dokonano porównania algorytmu ADASYN z algorytmem SMOTE pod względem poprawienia wydajności klasyfikatorów. Do porównania wykorzystano 3 różne niezbalasowane zbiory danych oraz 3 różne klasyfikatory. Stwierdzono, że obie metody wstępnego przetwarzania dają lepsze wyniki klasyfiakcji w większości przypadków. Porównując dwa algorytmy zauważono, że istnieje wzorzec, który przemawia na korzyść SMOTE podczas stosowania SVM do klasyfikacji. W większości przypadków różnice pomiędzy metodami ADASYN, a SMOTE są niezauważalne.

2.3 Borderline-SMOTE

Innym interesującym podejściem radzenia sobie z niezbalansowanymi zbiorami danych jest algorytm przedstawiony w artykule [4]. Algorytm borderline-SMOTE to modyfikacja algorytmu SMOTE, która ma na celu poprawę wydajności i jakości oversamplingu w przypadku, gdy w klasie mniejszości są obserwacje, które są odległe i pojawiają się w przestrzeni klasy większości.

Idea algorytmu borderline-SMOTE polega na tym, że zamiast tworzyć sztuczne przykłady tylko na podstawie mniejszościowej klasy, algorytm analizuje każdy przykład z klasy granicznej i decyduje, czy dany przykład powinien zostać "rozszerzony" poprzez stworzenie sztucznego przykładu. Jeśli przykład znajduje się blisko klasy mniejszościowej, to algorytm tworzy nowy sztuczny przykład na podstawie najbliższych sąsiadów z klasy mniejszościowej. W praktyce, algorytm borderline-SMOTE zaczyna od wyszukania granicznych próbek z klasy mniejszościowej. Próbki te mogą znajdować się wewnątrz klasy mniejszościowej, ale także w pobliżu klasy większościowej lub na granicy pomiędzy klasami. Dla każdej wybranej próbki granicznej, algorytm znajduje k najbliższych sąsiadów z tej samej klasy, a następnie wybiera losowo jednego z k sąsiadów i generuje nową syntetyczną próbkę przez interpolację między próbką graniczną a wybranym sąsiadem. Algorytm powtarza te operacje dla wszystkich próbek granicznych, aż do uzyskania zadanej liczby syntetycznych próbek.

W ten sposób, algorytm borderline-SMOTE pozwala na stworzenie sztucznych przykładów tylko tam, gdzie jest to naprawdę potrzebne, czyli dla przykładów z klas granicznych. Dzięki temu, algorytm ten może przynieść lepsze wyniki niż standardowy algorytm SMOTE. Eksperymenty przeprowadzone w artykule [4] pokazują, że metoda ta osiąga lepszy współczynnik TP oraz F-value niż sam algorytm SMOTE.

2.4 Random oversampling

Random oversampling to technika oversamplingu, która polega na losowym duplikowaniu obserwacji z mniejszej klasy, aby wyrównać liczbę obserwacji w obu klasach. Ta technika jest stosunkowo prosta w implementacji i najmniej zaawansowana spośród wymienionych metod oversamplingu niezbalansowanych danych oraz nie wymaga dużych zasobów obliczeniowych.

Wadą algorytmu Random oversampling jest to, że powielanie przypadków z klasy mniejszościowej może prowadzić do nadmiernego dopasowania modelu do danych, ponieważ tworzone są dokładne kopie przykładów klasy mniejszościowej. Ponadto technika ta może wprowadzić dużo większe wyzwania obliczeniowe dla klasyfikatorów, jeśli zestaw danych jest już dużego rozmiaru przed samym oversamplingiem.

W artykule [10] autorzy przeprowadzili porównanie techniki losowego oversamplingu z losowym undersamplingiem. Testów dokonano na 5 niezbalansowanych zbiorach danych. Jako klasyfikator został wykorzystany SVM, najpierw na oryginalnych niezrównoważonych zbiorach danych, a następnie na zestawach danych wygnerowanymi metodami losowego nadpróbkowania i podpróbkowania. Jako miarę poprawności klasyfikacji użyto średnią geometryczną. Na podstawie przeprowadzonych testów stwierdzono, że Random oversampling może skutkować lepszymi ogólnymi wynikami klasyfikacji niż Random undersampling.

2.5 SPIDER

Kolejnym ciekawym algorytmem oversamplingu jest SPIDER opisany w artykule [6]. Jest to algorytm wykonujący selektywny preprocessing danych niezbalansowanych. Wykorzystuje on charakterystykę próbek i na podstawie ich k najbliższych sąsiadów, dzieli je na dwie kategorie - bezpieczne (ang. safe) oraz zaszumione (ang. noisy), a następnie flaguje je. Przykłady ocenione jako bezpieczne powinny być również później poprawnie sklasyfikowane, natomiast próbki ocenione jako zaszumione mogą powodować błędy podczas klasyfikacji, zatem wymagają specjalnego przetworzenia. W kolejnym kroku przykłady poddawane są różnym operacjom. Metoda SPIDER zachowuje wszystkie przykłady z klasy mniejszościowej niezależnie od flagi i powiela część z nich oraz jednocześnie usuwa część przykładów z klasy większościowej. Założeniem algorytmu jest usuwanie tylko przykładów zaszumionych i zachowanie wszystkich przykładów bezpiecznych klasy większościowej.

Autorzy artykułu [6] zaproponowali trzy wersje algorytmu SPIDER: słabe wzmocnienie (ang. weak amplification), słabe wzmocnienie i ponowne etykietowanie (ang. weak amplification and relabeling) oraz silne wzmocnienie (ang. strong amplification). Pierwsza z nich jest wersją najprostszą. Skupia się ona na dodawaniu kopii przykładów zaszumionych z klasy mniejszościowej. Liczba kopii zależna jest od liczby bezpiecznych przykładów klasy większościowej w najbliższym sąsiedztwie punktu. Powoduje to wzmocnienie przykładów klasy mniejszościowej, co zwiększa ich szanse na wychwycenie przez klasyfikatory.

Druga wersja algorytmu SPIDER jest rozszerzeniem wersji pierwszej o ponowne etykietowanie. Kopiuje ona zaszumione przykłady z klasy mniejszościowej biorąc pod uwagę ich sąsiedztwo, ale także zmienia etykiety przykładów zaszumionych z klasy większościowej, które znajdują się w sąsiedtwie zaszumionych przykładów z klasy mniejszościowej. Przykłady te zostają przeetykietowane na klasę mniejszościową. Wersja ta wykazuje podobieństwo do metody SMOTE, lecz tworzy nowe przykłady za pomocą zmiany etykiet, a nie generowania sztucznych przykładów.

Silne wzmocnienie jest najbardziej zaawansowaną wersją algorytmu SPIDER. Kopiuje ona zarówno przykłady bezpieczne, jak i zaszumione z klasy mniejszościowej w zależności od ich najbliższego sąsiedztwa. Przykłady bezpieczne są kopiowane na podstawie ilości przykładów biezpiecznych z klasy większościowej w ich sąsiedztwie. Następnie przykłady zaszumione są klasyfikowane ponownie z użyciem rozszerzonego sąsiedztwa (5 sąsiadów, przy czym domyślne sąsiedztwo używane wcześniej i w innych wersjach algorytmu to 3). Przykłady ponownie sklasyfikowane poprawnie zostają kopiowane na podstawie liczby bezpiecznych przykładów klasy większościowej z ich najbliższego regularnego sąsiedztwa. Natomiast przykłady ponownie sklasyfikowane niepoprawnie są kopiowane na podstawie liczby bezpiecznych przykładów klasy większościowej z ich rozszerzonego sąsiedztwa.

3 Wybrane klasyfikatory

Klasyfikatory to algorytmy uczenia maszynowego używane do przypisywania danym wejściowym etykiet na podstawie zestawu zdefiniowanych wcześniej kategorii. Algorytmy te uczą się przewidywania do jakiej klasy należeć będą dane wejściowe na podstawie zbioru danych treningowych składającego się z przykładowych danych wejściowych i odpowiadających im danych wyjściowych.

3.1 Drzewa decyzyjne

Jednymi z najczęściej używanych klasyfikatorów są drzewa decyzyjne, czyli algorytmy będące nieparametryczną metodą uczenia maszynowego. Stosuje się je do klasyfikacji i regresji. Przewidują one wartości zmiennych z wykorzystaniem reguł decyzyjnych określonych na podstawie cech danych. Im bardziej złożone są reguły tym skuteczniejszy jest model. Zaletami drzew decyzyjnych są możliwość wizualizacji, łatwość interpretacji, a także niewielkie wymagania w zakresie przygotowania danych. Natomiast wadą jest możliwość powstania drzew niestabilnych lub zbyt złożonych, co spowoduje błędne klasyfikowanie danych [12].

3.2 K-Najbliższych Sąsiadów

Innym przykładem klasyfikatora jest K-Najbliższych Sąsiadów. Klasyfikator ten przeszukuje zbiór treningowy dla wybranej liczby k przykładów, które są najbliżej danego punktu i klasyfikuje go na podstawie klasy większości wśród jego k najbliższych sąsiadów. Wartość k jest zazwyczaj mała, a jej dokładny wybór zależy od klasyfikowanych danych. Zazwyczaj większa wartość k zmniejsza wpływ szumu na wynik klasyfikacji, lecz sprawia, że jej granice stają się mniej wyraźne [8].

3.3 Naiwny klasyfikator Bayesa

Kolejnym wybranym klasyfikatorem jest naiwny klasyfikator Bayesa. Jest to algorytm uczenia nadzorowanego wykorzystujący twierdzenie Bayesa. Zakłada on,

że wszystkie atrybuty wejściowe są niezależne od siebie i wpływ jednego atrybutu na klasę nie zależy od pozostałych, stąd nazywany jest naiwnym. Klasyfikacja za jego pomocą polega na wykonaniu serii obliczeń probabilistycznych w celu znalezienia najlepiej dopasowanej klasyfikacji dla wybranych danych. Zaletą tego klasyfikatora jest wysoka dokładność oraz skalowalność nawet w przypadku bardzo dużych zbiorów danych [3].

4 Cele eksperymentu

Celem eksperymentu jest porównanie skuteczności trzech różnych podejść do klasyfikacji danych niezbalansowanych: z użyciem metod preprocessingu takich jak SPIDER i SMOTE oraz klasyfikacji bez preprocessingu. W ramach eksperymentu zostanie przeprowadzona analiza wyników klasyfikacji, aby ocenić ich efektywność.
Głównymi celami eksperymentu są:

- Porównanie skuteczności klasyfikacji z użyciem metody SPIDER i klasyfikacji bez preprocessingu.
- Porównanie skuteczności klasyfikacji z użyciem metody SMOTE i klasyfikacji bez preprocessingu.
- Porównanie skuteczności klasyfikacji z użyciem metody SPIDER i klasyfikacji z użyciem metody SMOTE.
- Porównanie trzech wersji metody SPIDER między sobą.

Przeprowadzenie powyższych badań pozwoli na dokładne porównanie skuteczności klasyfikacji przy użyciu różnych metod preprocessingu i wybranie najlepszego podejścia dla danego problemu.

4.1 Research questions

1. W jakim stopniu preprocessing danych z użyciem każdej z dwóch metod wpłynie na efektywność klasyfikacji?
2. Czy pomimo poprawy wartości metryki Recall po zastosowaniu preprocessingu zachowana będzie zadowalająca wartość Specificity?
3. Który z algorytmów będzie powodował najmniejsze zmiany w rozkładzie klas?
4. Czy któraś z metod preprocessingu okaże się "lepsza" od innych czy ich efekty bedą porównywalne?

5 Plan eksperymentu

Eksperyment zostanie przeprowadzony na czternastu zbiorach danych z wykorzystaniem opisanych wcześniej klasyfikatorów: drzewa decyzyjne, K-Najbliższych Sąsiadów oraz naiwny klasyfikator Bayesa. Użyte zostaną dwie metody preprocessingu danych SMOTE oraz SPIDER w trzech wersjach. Parametr *k_neighbors*

w algorytmie SMOTE ustawiono na 3 (zamiast standardowych 5) z powodu użycia zbiorów danych glass5 oraz glass-0-1-6_vs_5. Te dwa konkretne zbiory cechują się bardzo małą ilością próbek klasy mniejszościowej. Podczas użycia 5-foldowej crosswalidacji z dwoma splitami zdarza się sytuacja, kiedy nie mamy nawet pięciu próbek klasy mniejszościowej w jednym foldzie. Na przykład glass5 posiada 214 wszystkich próbek, a jego IR wynosi 22.78, co oznacza, że mamy tylko 9 próbek mniejszościowych, więc po splicie w crosswalidacji otrzymujemy liczbę mniejszą od 5.

Do przeprowadzenia eksperymentu zastosowana zostanie procedura walidacji krzyżowej z pięcioma iteracjami z dwoma splitami. Ocena badań zostanie oparta o metryki opisane w dalszej części pracy: Recall, Specifity, Balanded Accuracy, Geometric Mean, Precision i F-measure.

5.1 Opis środowiska eksperymentalnego

Do przeprowadzenia eksperymentu wykorzystaną zostane następujące narzędzia:

- Python 3.10 z pakietami NumPy, Scikit-learn, Stream-learn, SciPy oraz Imbalanced-learn do przetwarzania danych i uczenia maszynowego.
- Pakiet Plotly do wykonywania wizualizacji wyników.
- PyCharm jako środowisko do przeprowadzenia eksperymentów i wizualizacji wyników.
- Jupyter Notebook jako środowisko do wizualizacji rozkładu danych oraz testowania algorytmu SPIDER.

5.2 Wybrane zestawy danych

Eksperyment zostanie przeprowadzony na 14 zbiorach danych pochodzących z repozytorium KEEL [13]. Wybrano 7 zbiorów danych o niskim współczynniku niezrównoważenia (ang. Imbalance ratio), czyli z przedziału (1.5-9] oraz 7 zbiorów danych o wysokim współczynniku niezrównoważenia z przedziału $(9, \infty)$. Liczba klas wszystkich we wszystkich wybranych zbiorach wynosi 2. Nazwy wybranych zbiorów wraz z ich parametrami takimi jak liczba cech, liczba instacji oraz Imbalance ratio zamieszczono w tabelach poniżej.

Tabela 1. Wybrane zbiory o współczynniku niezrównoważenia 1.5-9

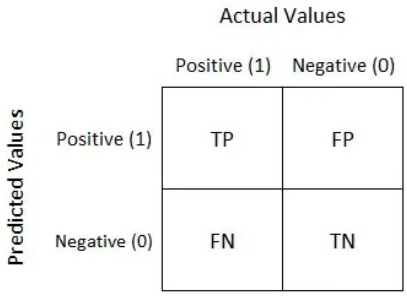
Nazwa zbioru	Liczba cech	Liczba instancji	Imbalance ratio
glass1	9	214	1.82
yeast1	8	1484	2.46
ecoli1	7	336	3.36
ecoli2	7	336	5.46
glass6	9	214	6.38
yeast3	8	1484	8.10
ecoli3	7	336	8.60

Tabela 2. Wybrane zbiory o współczynniku niezrównoważenia większym niż 9

Nazwa zbioru	Liczba cech	Liczba instancji	Imbalance ratio
yeast-2_vs_4	8	514	9.08
glass2	9	214	11.59
ecoli4	7	336	15.80
glass-0-1-6_vs_5	9	184	19.44
glass5	9	214	22.78
yeast4	8	1484	28.10
yeast6	8	1484	41.44

5.3 Ewaluacja eksperymentu

Ocena wyników eksperymentu zostanie przeprowadzona na podstawie wybranych metryk, czyli parametrów pokazujących skuteczność przeprowadzonej klasyfikacji. Wzory pozwalające na obliczenie metryk korzystają z Confusion Matrix, czyli tabeli pokazującej wszystkie kombinacje wartości przewidywanych przez model i rzeczywistych [11]. Rysunek przedstawiający Confusion Matrix zamieszczono poniżej:



Rysunek 1. Confusion Matrix. Źródło: [11]

Recall (sensitivity) pokazuje jaka część danych z klasy mniejszościowej została poprawnie sklasyfikowana. Wartośc tego parametru powinna być możliwie najwyższa, czyli jak najbardziej zbliżona do 1 [11]. Wylicza się go ze wzoru:

$$Recall = \frac{t_p}{t_p + f_n} \tag{1}$$

Specificity pokazuje jaka część danych z klasy większościowej została poprawnie sklasyfikowana. Jest to parametr analogiczny do Recall. Jego wzór to:

$$Specificity = \frac{t_n}{t_n + f_p} \tag{2}$$

Accuracy jest ogólną miarą dokładności klasyfikacji. Jej wartość może być niedokładna w przypadku danych niezbalansowanych, dlatego lepszym wyborem jest metryka Balanced accuracy, opisanan poniżej. Wzór na Accuracy to:

$$Accuracy = \frac{t_p + t_n}{t_p + t_n + f_p + f_n} \tag{3}$$

Balanced accuracy pokazuje jaka część danych z obu klas została poprawnie sklasyfikowana. Działa prawdłowo również w przypadku danych niezbalansowanych [7]. Jest to średnia z parametrów Recall i Specificity:

$$Balanced - accuracy = \frac{Recall + Specificity}{2} \tag{4}$$

Geometric mean (g-mean) mierzy równowagę pomiędzy wydajnością klasyfikacji dla klas większościowych i mniejszościowych. Jeśli wynik jest niski, wskazuje to na słabą wydajność klasyfikacji [7]. Jest to pierwiastek kwadratowy z iloczynu recall i specificity:

$$G - mean = \sqrt{Recall * Specificity} \tag{5}$$

Precision jest miarą dokładności modelu. Pokazuje ona jaką część danych sklasyfikowanych jako pozytywne stanowią te, które zostały prawidłowo sklasyfikowane [11]. Parametr ten wylicza się ze wrozu:

$$Precision = \frac{t_p}{t_p + f_p} \tag{6}$$

F-measure (F1 score) to średnia harmoniczna z Precision i Recall. Parametr ten jest szczególnie przydatny przy porównywaniu dwóch modeli, w których wartości Precision i Recall są skrajnie różne. Najlepsza możliwa wartość tej metryki to 1, a najgorsza to 0 [7]. Wzór na F1 score to:

$$F - measure = \frac{2 * Precision * Recall}{Precision + Recall} \tag{7}$$

5.4 Analiza statystyczna

W celu sprawdzenia czy otrzymane dla poszczególnych zbiorów danych i metod preprocessingu wartości metryki Balanced accuracy istotnie różnią się od siebie wykorzystany zostanie Test t-Studenta. Test ten wskaże, czy otrzymane w wynikach różnice są przypadkowe, czy też istotne statystycznie. Dzięki temu możliwe będzie porównanie efektywności metod preprocessingu oraz wskazanie, które istotnie poprawiają jakość klasyfikacji dla kolejnych zbiorów danych. Ponadto dla każdego zbioru danych zostanie wykonane rangowanie wyników dla

kolejnych metod preprocessingu. Otrzymane rangi zostaną uśrednione i wykonany zostanie test Wilcoxona. Test ten pozwoli ocenić średnią efektywność metod preprocessingu dla wszystkich badanych zbiorów danych. W obu testach przyjęte zostaną następujące hipotezy:

- H0: Różnice pomiędzy otrzymanymi wynikami nie są znaczące.
- H1: Różnice pomiędzy otrzymanymi wynikami są znaczące.

Wykorzystana zostanie wartość graniczna p-value równa 0,05. Różnice w wynikach z wyższą wartością p-value zostaną uznane za nieistotne statystycznie. Natomiast dla tych z niższą wartością p-value zostanie odrzucona hipoteza zerowa i przyjęta hipoteza alternatywna, więc różnice będą uznane za istotne statystycznie. W tych przypadkach możliwe będzie wskazanie lepszej i gorszej metody preprocessingu danych.

6 Wyniki eksperymentu

W tym rozdziale przedstawione zostaną rezulaty ekperymentu. Są one podzielone na trzy części, gdyż kolejno pokazano wyniki dla każdego wykorzystanego klasyfikatora: drzewa decyzyjne, K-Najbliższych Sąsiadów oraz naiwny klasyfikator Bayesa. W tabelach oraz na wykresach zastosowano następujące skróty odpowiadające wykorzystanym metodom preprocessingu danych:

- none - brak preprocessingu
- smote - metoda SMOTE
- spiderW - metoda SPIDER w wersji "weak amplification"
- spiderWR - metoda SPIDER w wersji "weak amplification and relabeling"
- spiderS - metoda SPIDER w wersji "strong amplification"

W tabelach 3-5 przedstawiono dla kolejnych klasyfikatorów uśrednione wartości metryki Balanced accuracy dla każdego zbioru danych przy użyciu kolejnych metod preprocessingu. W ostatnim wierszu tabeli znajdują się wartości średnie metryki Balanced Accuracy uzyskane na podstawie wyników dla wszystkich zbiorów dla danej metody preprocessingu oraz średnie rangi przyznane w celu wykonania testu Wilcoxona. Im wyższa ranga tym wynik został lepiej oceniony.

W każdym wierszu tabel 3-5 pod wynikami znajdują się liczby od 1 do 5 lub wykreślone pola. Liczby te to numery kolumn zawierających metody statystycznie istotnie gorsze dla danego zbioru danych. Zatem jeśli przykładowo pod drugą z kolei metodą znajduje się numer 1, oznacza to, że metoda druga jest statystycznie lepsza od metody pierwszej, czyli w tym przypadku braku preprocessingu. W ostatnim wierszu liczby te są ogólną oceną średnich rang na tej samej zasadzie. Wykreślone pola oznaczają natomiast, że dana metoda nie jest istotnie statystycznie lepsza od żadnej z pozostałych w danym przypadku.

Na wykresach radarowych przedstawionych na rysunkach 2-4 pokazano natomiast uśrednione dla wszystkich zbiorów danych i metod preprocessingu wartości wszystkich wybranych metryk.

6.1 Drzewa decyzyjne - wyniki

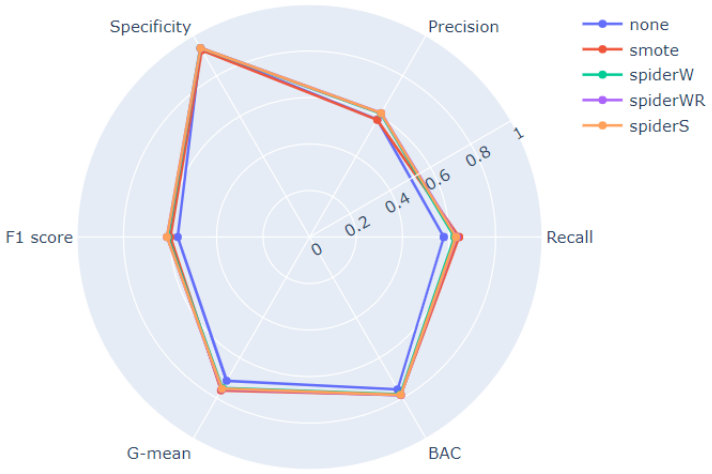
W tabeli 3 przedstawiono wyniki metryki Balanaced accuracy uzyskane dla klasyfikatora drzew decyzyjnych.

Tabela 3. Wartości Balanced accuracy dla klasyfikatora drzew decyzyjnych

Nazwa zbioru	none	smote	spiderW	spiderWR	spiderS
glass1	0.7195	0.6976	0.7176	0.7141	0.7210
	—	—	—	—	—
yeast1	0.6331	0.6445	0.6536	0.6521	0.6530
	—	—	1	1	1
ecoli1	0.8230	0.8227	0.8433	0.8388	0.8445
	—	—	—	—	—
ecoli2	0.8340	0.8625	0.8762	0.8794	0.8799
	—	—	1	1	1
glass6	0.8892	0.8982	0.8874	0.8896	0.8940
	—	—	—	—	—
yeast3	0.8227	0.8417	0.8687	0.8683	0.8665
	—	—	1,2	1,2	1,2
ecoli3	0.7122	0.7265	0.7504	0.7651	0.7692
	—	—	—	1	1
yeast-2_vs_4	0.8455	0.8645	0.8541	0.8567	0.8578
	—	—	—	—	—
glass2	0.5014	0.6257	0.5639	0.5784	0.5596
	—	1	1	1	1
ecoli4	0.8192	0.8036	0.8155	0.8011	0.8011
	—	—	—	—	—
glass-0-1-6_vs_5	0.7901	0.9130	0.8805	0.9022	0.8517
	—	1	—	—	—
glass5	0.8247	0.8587	0.8207	0.8507	0.8721
	—	—	—	—	—
yeast4	0.6415	0.6740	0.6690	0.6666	0.6689
	—	—	—	—	—
yeast6	0.7351	0.7562	0.7291	0.7277	0.7312
	—	—	—	—	—
Średnie BAC	0.7565	0.7850	0.7807	0.7851	0.7836
Średnia ranga	1.9286	3.3571	3.0714	3.0000	3.6428
	—	1	1	1	1

Wyniki przedstawione w tabeli 3 uzyskane dla klasyfikatora drzew decyzyjnych wykazują ogólnie dla wszystkich wybranych zbiorów danych, że zastosowanie metod preprocessingu SMOTE lub SPIDER we wszystkich wersjach jest statystycznie lepsze od braku preprocessingu danych. Średnie różnice pomiędzy metodami SPIDER i SMOTE są nieistotne statystycznie. Dla poszczególnych zbiorów danych widać natomiast, że w większości przypadków (dla 8 z 14 zbiorów) żadna z metod preprocessingu statystycznie istotnie nie poprawia rezulta-

tów klasyfikacji. Dla pozostałych zbiorów danych w kilku przypadkach (yeast1, ecoli2, ecoli3) najlepsze okazały się poszczególne wersje metody SPIDER, a dla zbioru danych glass-0-1-6_vs_5 najlepsze rezultaty uzyskała metoda SMOTE. Dla zbioru danych yeast3 metoda SPIDER we wszystkich wersjach zadziała statystycznie istotnie lepiej niż brak preprocessingu oraz metoda SMOTE. Brak preprocessingu nie przyniósł istotnie statystycznie lepszych wyników niż metody SPIDER i SMOTE w żadnym przypadku.



Rysunek 2. Uśrednione metryki dla klasyfikatora drzew decyzyjnych

Na wykresie z rysunku 2 widać, że wszystkie metody preprocessingu osiągnęły podobne rezulaty dla wszystkich metryk, jednak brak preprocessingu wypadł najgorzej dla 4 z nich. Wyniki otrzymane dla metody SPIDER nie różnią się znacząco od wyników dla metody SMOTE. Wyniki z użyciem preprocessingu metodą SMOTE wypadają najlepiej dla 3 z 6 metryk, podobnie wyniki uzyskane dla metody SPIDER. Wszystkie wersje metody SPIDER osiągnęły bardzo zbliżone do siebie nawzajem wyniki, gdyż punkty dotyczące kolejnych wersji tych metod na wykresie prawie się pokrywają.

6.2 K-Najbliższych Sąsiadów - wyniki

Wyniki przedstawione w tabeli 4 uzyskane dla klasyfikatora K-Najbliższych Sąsiadów wykazują ogólnie dla wszystkich wybranych zbiorów danych, że zastosowanie metod preprocessingu SMOTE lub SPIDER we wszystkich wersjach jest statystycznie lepsze od braku preprocessingu danych. Średnie różnice pomiędzy metodami SPIDER i SMOTE są nieistotne statystycznie. Dla poszczególnych zbiorów danych widać podobne rezultaty. Dla 12 z 14 zbiorów wyniki są takie

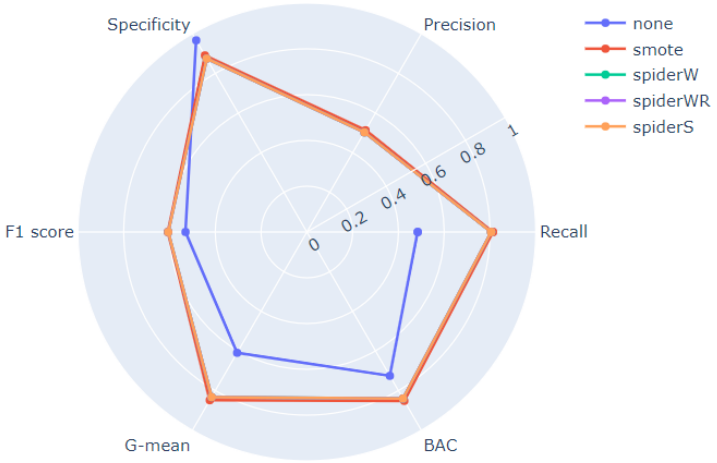
same jak dla ogółu, czyli zastosowanie metod SMOTE lub SPIDER jest statystycznie lepsze od braku preprocessingu. Dla pozostałych dwóch zbiorów (ecoli2 i ecoli4) żadna z metod preprocessingu statystycznie istotnie nie poprawia rezultatów klasyfikacji. W żadnym przypadku brak preprocessingu nie przyniósł istotnie statystycznie lepszych wyników niż metody SPIDER i SMOTE. Metody te nie wykazały również istotnych statystycznie różnic pomiędzy sobą dla żadnego zbioru danych.

Tabela 4. Wartości Balanced accuracy dla klasyfikatora K-Najbliższych Sąsiadów

Nazwa zbioru	none	smote	spiderW	spiderWR	spiderS
glass1	0.7281	0.7690	0.7776	0.7776	0.7776
	—	1	1	1	1
yeast1	0.6487	0.6813	0.6804	0.6804	0.6804
	—	1	1	1	1
ecoli1	0.8284	0.8768	0.8843	0.8843	0.8843
	—	1	1	1	1
ecoli2	0.9142	0.9182	0.9070	0.9070	0.9070
	—	—	—	—	—
glass6	0.8690	0.9069	0.9089	0.9089	0.9089
	—	1	1	1	1
yeast3	0.8228	0.8857	0.8872	0.8872	0.8872
	—	1	1	1	1
ecoli3	0.7690	0.8581	0.8604	0.8604	0.8604
	—	1	1	1	1
yeast-2_vs_4	0.8195	0.8917	0.8727	0.8727	0.8727
	—	1	1	1	1
glass2	0.5273	0.7490	0.7378	0.7378	0.7378
	—	1	1	1	1
ecoli4	0.8825	0.9242	0.9245	0.9245	0.9245
	—	—	—	—	—
glass-0-1-6_vs_5	0.5299	0.9007	0.8574	0.8574	0.8574
	—	1	1	1	1
glass5	0.5085	0.9298	0.8638	0.8638	0.8638
	—	1	1	1	1
yeast4	0.5667	0.8004	0.7581	0.7581	0.7581
	—	1	1	1	1
yeast6	0.7427	0.8334	0.8394	0.8394	0.8394
	—	1	1	1	1
Średnie BAC	0.7255	0.8518	0.8400	0.8400	0.8400
Średnia ranga	1.2143	3.5000	3.4286	3.4286	3.4286
	—	1	1	1	1

Na wykresie z rysunku 3 widać, że metody preprocessingu SMOTE oraz SPIDER we wszystkich wersjach uzyskały bardzo zbliżone rezulaty dla wszystkich metryk, minimalnie lepiej wypadła metoda SMOTE. Wszystkie wersje metody

SPIDER osiągnęły dokładnie takie same wyniki, punkty dotyczące tych metod na wykresie pokrywają się. Znacznie gorzej niż metody SMOTE i SPIDER wypadł brak preprocessingu. Ponadto niemożliwe okazało się uzyskanie wyników metryki Precision dla braku preprocessingu przy zastosowaniu klasyfikatora K-Najbliższych Sąsiadów. Brak preprocessingu uzyskał jednak najwyższą wartość metryki Specificity.



Rysunek 3. Uśrednione metryki dla klasyfikatora K-Najbliższych Sąsiadów

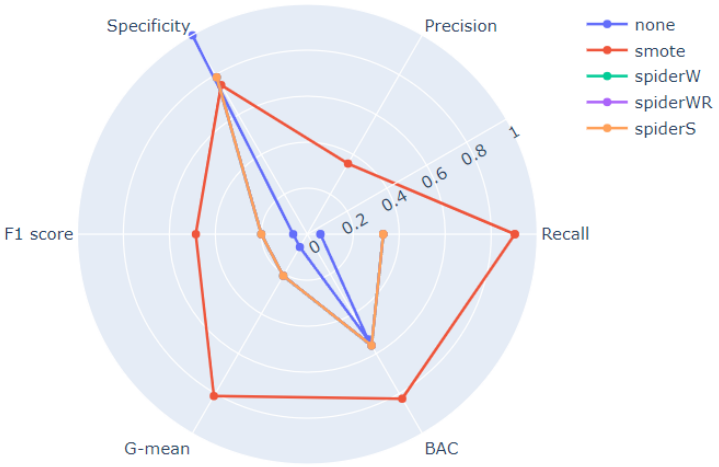
6.3 Naiwny klasyfikator Bayesa - wyniki

Wyniki przedstawione w tabeli 5 uzyskane dla naiwnego klasyfikatora Bayesa wykazują ogólnie dla wszystkich wybranych zbiorów danych, że zastosowanie metody SMOTE jest statystycznie istotnie najlepsze ze wszystkich badanych możliwości. Zastosowanie metody SPIDER we wszystkich wersjach jest natomiast istotnie lepsze tylko od braku preprocessingu. Dla poszczególnych zbiorów widać podobne rezultaty dla metody SMOTE. Dla 12 z 14 zbiorów danych uzyskała ona istotnie statystycznie najlepsze wyniki. Dla 6 zbiorów (w tym 5 wspomnianych wcześniej) również SPIDER we wszystkich wersjach okazał się lepszy od braku preprocessingu danych. Dla zbioru glass1 SMOTE nie uzyskał wyników istotnie lepszych od żadnej wersji metody SPIDER ani też odwrotnie SPIDER nie okazał się lepszy od SMOTE, jednak wszystkie te metody uzyskały wyniki istotnie lepsze od braku preprocessingu. Natomiast dla zbioru glass6 jedynie brak preprocessingu jest staystycznie istotnie lepszy od zastosowania metody SMOTE. Dla tego zbioru uzyskano też ogólnie znacznie lepsze wyniki Balanced accuracy przy braku preprocessingu i użyciu metody SPIDER niż dla pozostałych zbiorów danych.

Tabela 5. Wartości Balanced accuracy dla naiwnego klasyfikatora Bayesa

Nazwa zbioru	none	smote	spiderW	spiderWR	spiderS
glass1	0.4962	0.5780	0.5786	0.5786	0.5786
	—	1	1	1	1
yeast1	0.5000	0.6764	0.5071	0.5071	0.5071
	—	1,3,4,5	1	1	1
ecoli1	0.5000	0.8611	0.5255	0.5255	0.5255
	—	1,3,4,5	1	1	1
ecoli2	0.5000	0.8753	0.5115	0.5115	0.5115
	—	1,3,4,5	1	1	1
glass6	0.8941	0.8590	0.8884	0.8884	0.8884
	2	—	—	—	—
yeast3	0.5000	0.8291	0.5000	0.5000	0.5000
	—	1,3,4,5	—	—	—
ecoli3	0.5000	0.8636	0.5000	0.5000	0.5000
	—	1,3,4,5	—	—	—
yeast-2_vs_4	0.5000	0.8748	0.5000	0.5000	0.5000
	—	1,3,4,5	—	—	—
glass2	0.5000	0.6561	0.4954	0.4954	0.4954
	—	1,3,4,5	—	—	—
ecoli4	0.5000	0.8861	0.5000	0.5000	0.5000
	—	1,3,4,5	—	—	—
glass-0-1-6_vs_5	0.4994	0.9549	0.6720	0.6720	0.6720
	—	1,3,4,5	1	1	1
glass5	0.5000	0.9522	0.6497	0.6497	0.6497
	—	1,3,4,5	1	1	1
yeast4	0.5000	0.8298	0.5000	0.5000	0.5000
	—	1,3,4,5	—	—	—
yeast6	0.5000	0.8647	0.5000	0.5000	0.5000
	—	1,3,4,5	—	—	—
Średnie BAC	0.5278	0.8258	0.5592	0.5592	0.5592
Średnia ranga	2.1428	4.5000	2.7857	2.7857	2.7857
	—	1,3,4,5	1	1	1

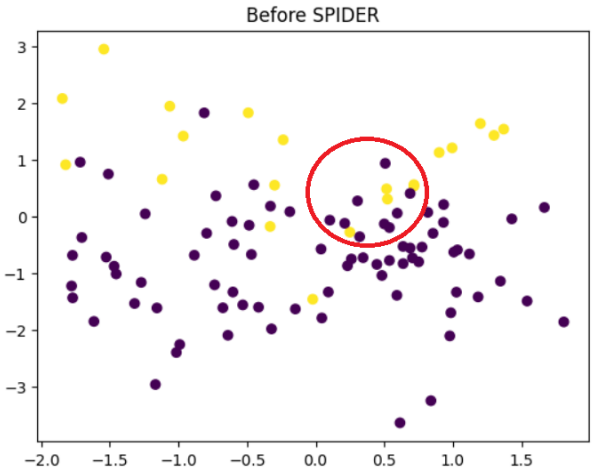
Na wykresie z rysunku 4 widać, że jedynie dla metody SMOTE udało się uzyskać wyniki dla wszystkich metryk przy użyciu naiwnego klasyfikatora Bayesa. Dla metody SPIDER oraz w przypadku braku preprocessingu niemożliwe okazało się uzyskanie wyników metryki Precision. Zdecydowanie najlepsze wyniki wszystkich metryk z wyjątkiem Specificity uzyskała metoda SMOTE. Następna w kolejności, choć znacznie gorsza, okazała się metoda SPIDER we wszystkich wersjach. Wszystkie wersje tej metody osiągnęły dokładnie takie same wyniki, punkty dotyczące tych metod na wykresie pokrywają się. Najgorzej w tym porównaniu wypadł natomiast brak preprocessingu. Uzyskał on jednak najlepszy wynik metryki Specificity - zbliżony do 1. Za to wynik parametru Recall dla braku preprocessingu jest zbliżony do 0.



Rysunek 4. Uśrednione metryki dla naiwnego klasyfikatora Bayesa

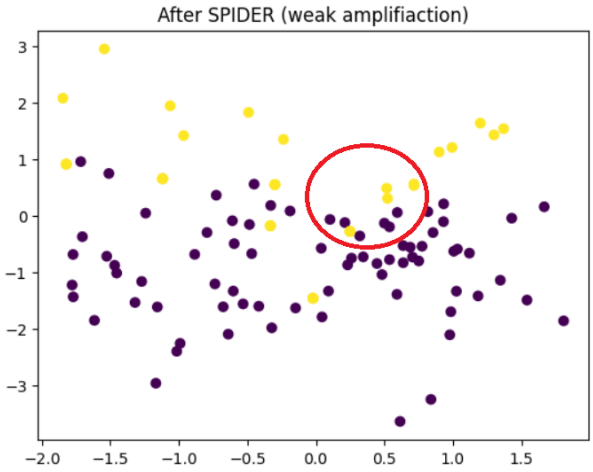
6.4 Wizualizacja działania preprocessingu metodą SPIDER

W celu przyjrzenia się co dokładnie algorytm SPIDER robi z danymi utworzono sztuczny zbiór danych składający się z dwóch cech i 100 próbek łącznie, aby można było go przedstawić na płaszczyźnie. Rozkład klas został ustalony na 0,8 do 0,2. Zbiór ten pokazano na rysunku 5. Obszar, w którym pojawiają się zmiany zaznaczono czerwoną obwiednią na rysunkach 5-8.

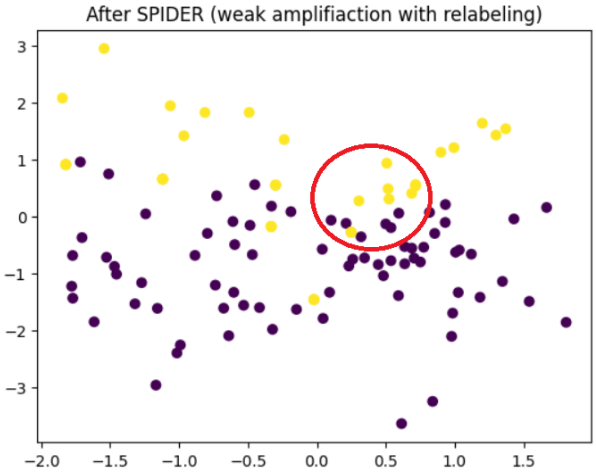


Rysunek 5. Oryginalny rozkład danych

Jak można zauważyć na rysunku 6 SPIDER w wersji "weak amplification" usuwa próbki z klasy większościowej sklasyfikowane przez algorytm jako "noisy". Natomiast w wersji "weak amplification and relabeling" pokazanej na rysunku 7 etykiety tych próbek zostają zmienione na klasę mniejszościową.

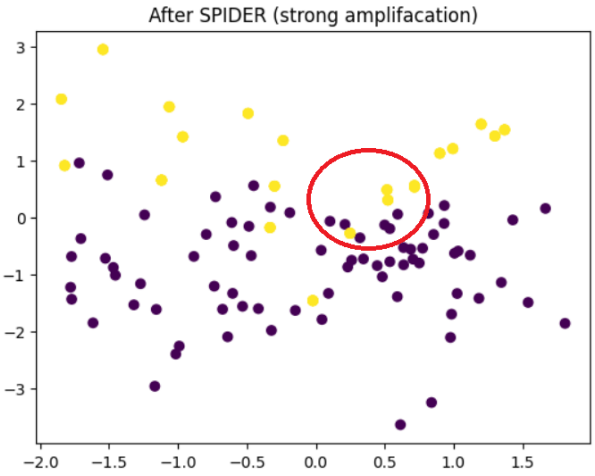


Rysunek 6. Rozkład danych po preprocessingu SPIDER w wersji "weak amplification"



Rysunek 7. Rozkład danych po preprocessingu SPIDER w wersji "weak amplification and relabeling"

Na żadnym z rysunków 6-8 nie widać natomiast samej amplifikacji, ponieważ próbki z klasy mniejszościowej są wzmacniane poprzez tworzenie ich identycznych kopii, dlatego punkty te nakładają się na siebie na wykresach.



Rysunek 8. Rozkład danych po preprocessingu SPIDER w wersji "strong amplification"

7 Podsumowanie

W tej sekcji przedstawione zostaną wnioski wyciągnięte na podstawie rezulatatów przeprowadzonego ekperymentu. Są one podzielone tematycznie na trzy części. Pierwsza z nich dotyczy wykorzystanych klasyfikatorów, druga obejmuje przetestowane metody preprocessingu, natomiast trzecia to pozostałe wnioski, niezaliczające się do żadnej z poprzednich grup.

7.1 Klasyfikatory - wnioski

Klasyfikator drzew decyzyjnych jest dobrym wyborem w przypadku problemu klasyfikacji danych niezbalansowanych, w szczególności przy braku zastosowania preprocessingu, ponieważ jest odporny na dane niezbalansowane. Rezulaty uzyskane przy jego pomocy dla większości wybranych zbiorów danych okazały się bardzo zbliżone, zarówno przy zastosowaniu metod preprocessingu SMOTE i SPIDER, jak i bez preprocessingu danych. Jedynie dla 6 z 14 zbiorów preprocessing jakąkolwiek z metod istotnie statystycznie poprawił wyniki klasyfikacji. Klasyfikator ten nie uzyskał jednak najlepszy wyników spośród wszystkich badanych klasyfikatorów.

Klasyfikator K-Najbliższych Sąsiadów wypadł najlepiej z testowanych klasyfikatorów w połączeniu ze stosowanymi metodami preprocessingu. Dla 12 z 14

zbiorów danych każda zastosowana metoda preprocessingu istotnie statystycznie poprawiła wyniki klasyfikacji w stosunku do braku preprocessingu.

Naiwnego klasyfikatora Bayesa nie jest dobrym wyborem dla problemu danych niezbalansowanych, gdy nie zostanie dobrana do niego odpowiednia metoda preprocessingu. Klasyfikator ten otrzymał zdecydowanie najgorsze wyniki w przypadku braku preprocessingu, ale także w przypadku użycia metody SPIDER we wszystkich wersjach. Jedynie metoda SMOTE w połączeniu w tym klasyfikatorem uzyskała wysokie wyniki metryki Balanced accuracy. W tym przypadku SMOTE statystycznie istotnie poprawił klasyfikację w stosunku do wszystkich pozostałych metod dla 12 z 14 zbiorów danych.

7.2 Metody preprocessingu - wnioski

Dla wybranych zbiorów danych oraz klasyfikatorów metoda SMOTE okazała się najlepsza. Uzyskała ona najwyższe rangi dla klasyfikatora K-Najbliższych Sąsiadów i naiwnego klasyfikatora Bayesa, a także drugą najlepszą, niegorszą istotsnie statystycznie od pierwszej, rangę dla klasyfikatora drzew decyzyjnych. Metoda ta istotnie poprawiła klasyfikację w stosunku do braku preprocessingu dla wszystkich klasyfikatorów, a także w stosunku do metody SPIDER dla naiwnego klasyfikatora Bayesa.

Metoda SPIDER we wszystkich wersjach najlepiej sprawdziła się w przypadku wykorzystania klasyfikatora K-Najbliższych Sąsiadów. Poprawiła ona w tym przypadku istotnie statystycznie efektywność klasyfikacji dla 12 z 14 zbiorów danych.

Wersje metody SPIDER nie różnią się od siebie nawzajem istotnie statystycznie w żadnym przypadku dla wybranych zbiorów i klasyfikatorów. Ponadto dla dla klasyfikatora K-Najbliższych Sąsiadów i naiwnego klasyfikatora Bayesa wszytskie wersje zwacają takie same wyniki dla tych samych zbiorów. Może to zależeć od wykorzystanego klasyfikatora, ponieważ jeśli kolejne wersje metody SPIDER bardzo mało zmienią w zbiorze treningowym to niektóre klasyfikatory wytrenują się identycznie niezależnie od wersji. Wtedy niewielkie różnice mogą zostać niewychwycone również w zbiorze testowym, przez co zwrócone zostaną identyczne wyniki.

Niestosowanie preprocessingu danych w przypadku klasyfikacji danych niezbalansowanych nie przynosi korzystnych efektów ogółem dla żadnego z wybranych klasyfikatorów i prawie dla żadnego badanego zbioru danych. Wyjątkiem jest zbiór glass6, dla którego przy użyciu naiwnego klasyfikatora Bayesa uzyskano statystycznie istotnie lepszy wynik dla braku preprocessingu w stosunku do metody SPIDER. Jednak w tym przypadku każda metoda uzyskała stosunkowo dobre wyniki. Wszystkie inne zbiory wypadły znacznie gorzej przy braku preprocessingu danych.

7.3 Pozostałe wnioski

Dla klasyfikatora K-Najbliższych Sąsiadów i naiwnego klasyfikatora Bayesa otrzymano bardzo wysokie, zbliżone do 1, wartości metryki Specificity dla braku pre-

processingu, ponieważ klasyfikacja w tym przypadku jest bardzo stronnicza. W związku ze znaczącą przewagą próbek klasy większościowej podczas trenowania, klasyfikatory te klasyfikują większość wszystkich próbek do klasy większościowej podczas testowania. Zwiększa się więc skuteczność klasyfikacji próbek prawidziwie należących do klasy większościowej, a zmniejsza się skuteczność klasyfikacji tych należących do klasy mniejszościowej. Stąd też wartość parametru Recall jest znacznie mniejsza, szczególnie w przypadku naiwnego klasyfikatora Bayesa jest ona zbliżona do 0.

Najmniejsze zmiany w rozkładzie klas powoduje algorytm SPIDER w wersji "weak amplification". Wynika to z tego, że wersja ta nie stosuje ponownego etykietowania ani nie ingeruje tak bardzo w klasę mniejszościową jak na przykład wersja "strong amplification", ponieważ nie kopiuje tak wielu przypadków.

Literatura

1. Alberto Fernandez, Salvador Garcia, Francisco Herrera, Nitesh V. Chawla, *"SMOTE for Learning from Imbalanced Data: Progress andChallenges, Marking the 15-year Anniversary"*. Journal of Artificial Intelligence Research 61, 2018.
2. Bartosz Krawczyk, *"Learning from imbalanced data: open challenges and future directions"*. Progress in Artificial Intelligence 5.4, 2016.
3. Feng-Jen Yang, *"An Implementation of Naive Bayes Classifier"*. 2018 International Conference on Computational Science and Computational Intelligence (CSCI), Las Vegas, NV, USA, 2018.
4. Hui Han, Wen-Yuan Wang, Bing-Huan Mao, *"Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning"*. Advances in Intelligent Computing. ICIC 2005. Lecture Notes in Computer Science, 2005.
5. Jakob Brandt, Emil Lanzén, *"A Comparative Review of SMOTE and ADASYN in Imbalanced Data Classification"*. Department of Statistics, Uppsala University, 2020.
6. Jerzy Stefanowski, Szymon Wilk, *"Selective Pre-processing of Imbalanced Data for Improving Classification Performance"*. Lecture Notes in Computer Science, 2008.
7. Josephine Akosa, *"Predictive accuracy: A misleading performance measure for highly imbalanced data"*. Proceedings of the SAS global forum. Vol. 12, 2017.
8. Pádraig Cunningham, Sarah Jane Delany, *"k-Nearest Neighbour Classifiers - A Tutorial"*. ACM Computing Surveys, Volume 54, Issue 6, 2022.
9. Roweida Mohammed, Jumanah Rawashdeh and Malak Abdullah, *"Machine Learning with Oversampling and Undersampling Techniques: Overview Study and Experimental Results"*. 2020 11th International Conference on Information and Communication Systems (ICICS), Irbid, Jordan, 2020.
10. Rukshan Batuwita, Vasile Palade, *"Efficient Resampling Methods for Training Support Vector Machines with Imbalanced Datasets"*. The 2010 International Joint Conference on Neural Networks, 2010.
11. Sarang Narkhede, *"Understanding confusion matrix"*. Towards Data Science 180.1, 2018.
12. Sotiris Kotsiantis, *"Decision trees: a recent overview"*. Artif Intell Rev 39, 261–283, 2013.
13. Joaquin Derrac, *"Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework"*. J. Mult. Valued Logic Soft Comput 17:2-3, 2015.