# Projekt ZUM

# Temat: Tworzenie zespołu modeli klasyfikacji przez wielokrotne stosowanie dowolnego algorytmu klasyfikacji do bootstrapowych prób ze zbioru trenującego z losowo wybranym podzbiorem atrybutów. Porównanie z algorytmami klasyfikacji tworzącymi pojedyncze modele oraz implementacjami algorytmów bagging i random forest dostępnymi w środowisku R lub Python

Dokumentacja

Autorzy: Jakub Szczepanowski, Oleh Hoba
Prowadzący: Stanisław Kozdrowski

## 1. Interpretacja tematu projektu

Idea modeli zespołowych polega na stworzeniu modelu będącego silnym klasyfikatorem na podstawie wielu składowych klasyfikatorów słabych (modele bazowe). Najbardziej popularnym, ale również skutecznym modelem tego typu jest las losowy. Las losowy to metoda uczenia zespołowego, w którym jak sama nazwa wskazuje stosuje się wiele drzew decyzyjnych do predykcji. Słowo „losowy'' w nazwie wskazuje na zastosowanie losowości w celu otrzymania różnych, słabszych modeli, które razem będą głosować do jakiej klasy należy przyporządkować dany przykład. Metody zespołowe charakteryzują się swobodą w zapewnianiu różnorodności modeli bazowych, dlatego można eksperymentować stosując różne algorytmy, zbiory trenujące itd. W tym projekcie zostaną zastosowane różne modele bazowe, które będą trenowane za pomocą różnych losowych podzbiorów zbioru uczącego powstałych poprzez losowanie próbek ze zwracaniem zwane agregacją. Do tego należy również dołożyć losowanie atrybutów dla każdego podzbioru, żeby osiągnąć jeszcze większą różnorodność. Ta druga opcja powinna zostać dodana jako hiperparametr regularyzacyjny estymatora. Liczba losowanych atrybutów może być różna, ale popularnym wyznacznikiem ilości jest zaokrąglona w dół wartość pierwiastka kwadratowego z liczby atrybutów. Drugą stosowaną techniką jest zlogarytmowanie liczby atrybutów z użyciem podstawy 2.

Niezbędnym z punktu widzenia modeli zespołowych elementem predyktora jest głosowanie. Może odbywać się ono na dwa różne sposoby. Jeżeli modele bazowe zwracają wynik wskazując na konkretną klasę użyte zostanie głosowanie twarde. W tym przypadku klasa, która otrzymała najwięcej głosów wygrywa. Natomiast jeżeli modele składowe są w stanie zwrócić wartość prawdopodobieństwa przynależności do danych klas zostanie użyte tzw. głosowanie miękkie. Dzięki technice głosowania miękkiego można uzyskać lepszą jakość, ponieważ zawiera się w nim miara pewności modelu za wskazaniem jakiejś konkretnej klasy. Samo głosowanie odbywa się poprzez uśrednianie prawdopodobieństw i zwracanie maksymalnego wyniku.

Elementem algorytmu uczenia maszynowego jest osłabianie modeli składowych za pomocą technik zwanych: agregacją bootstrapową, wklejaniem oraz stosowaniem podprzestrzeni losowych. Agregacja - jak to już zostało wspomniane - polega na losowaniu próbek ze zbioru trenującego ze zwracaniem, wklejanie natomiast na próbkowaniu bez zwracania. Zastosowanie podprzestrzeni losowej to po prostu wylosowanie atrybutów, do których ma być zawężone uczenie modelu bazowego.

# 2. Opis części implementacyjnej

Zakres prac projektowych zawęża się do zaimplementowania klasy modelu zespołowego, wraz z technikami agregacji bootstrapowej, głosowania oraz losowania atrybutów. Obiekty różnych technik klasyfikacji będą pochodziły z gotowej implementacji pochodzącej z pakietu Scikit-Learn. Cały projekt zostanie wykonany w języku Python. Następnym etapem będzie przeprowadzenie eksperymentów porównujących własną implementację z działaniem gotowej klasy stosującej bagging (agregowanie), lasu losowego oraz pojedynczego modelu.

# 3. Lista algorytmów, które będą wykorzystane w eksperymentach

- Bagging
- Logistic Regression (regresja logistyczna)
- Drzewo decyzyjne
- KNN (k-nearest neighbors)
- SVM (Support Vector Machines)
- Random Forest
- Naive Bayes
- Można też rozważyć wykorzystanie innych algorytmów klasyfikacji

# 4. Plan badań

W celu porównania skuteczności różnych algorytmów klasyfikacji i tworzenia zespołów modeli klasyfikacji:

1) Przeniesienie danych do preferowanych struktur takich jak ramka danych pakietu Pandas, czy struktura tablicowa pakietu NumPy
2) Wydzielenie etykiet
3) W razie potrzeby zmapowanie danych kategorycznych na dane numeryczne w zależności od ich charakterystyki (nominalne, rangowane)
4) Sprawdzenie, czy dane są kompletne oraz uzupełnienie ewentualnych wartości null wartością najczęstszej kategorii (dane kategoryczne) lub mediany (dane numeryczne)
5) Standaryzacja danych
6) Podział zbiorów danych na zbiór treningowy (80%) i zbiór testowy (20%)
7) Stworzenie pojedynczych modeli klasyfikacji za pomocą algorytmów:
    - Drzewo decyzyjne
    - KNN (k-nearest neighbors)

    - Logistic Regression (regresja logistyczna)
    - Naive Bayes
    - SVM (Support Vector Machines)
8) Stworzenie zespołów modeli klasyfikacji za pomocą algorytmów:
    - Bagging z wykorzystaniem powyższych algorytmów
    - Własnej implementacji z wykorzystaniem powyższych algorytmów
    - Random Forest
9) Porównanie skuteczności pojedynczych modeli klasyfikacji, zespołów modeli klasyfikacji na podstawie metryk:
    - Accuracy (dokładność)
    - Precision (precyzja)
    - Recall (czułość)
    - F1 score
    - macierz pomyłek
    - krzywa ROC wraz z AUC
10) Analiza wyników i wnioski.
11) Opcjonalnie, porównanie czasu trenowania i predykcji każdego algorytmu.

# 5. Zbiór danych

W celu przeprowadzenia eksperymentów można wykorzystać popularne zbiory danych, takie jak:

- Iris dataset - zbiór danych dotyczący trzech gatunków irysów (Setosa, Versicolor, Virginica), których pąki i płatki zostały zmierzone pod względem długości i szerokości
- Breast Cancer dataset - zbiór danych zawierający informacje o guzach piersi, w tym ich rozmiar, kształt, konsystencję i inne czynniki
- Wine dataset - zbiór danych dotyczący cech wina, takich jak zawartość alkoholu, kwasowość, pH itp., dla trzech różnych gatunków wina
- Pima Indians Diabetes dataset - zbiór danych zawierający dane diagnostyczne w celu wykrywania cukrzycy u pacjenta

# 6. Opis implementacji

Implementację klasy modelu zespołowego (EnsembleClassifier) wykorzystującą bagging oparto o pola przechowujące obiekt bazowy dowolnego algorytmu pochodzącego z pakietu sklearn, listę indeksów wylosowanych atrybutów (przy randomizacji cech), hiperparametry: ilość estymatorów, flagę randomizacji cech, metodę fit (estymator), metodę predict (predyktor klasy), predict_proba (predyktor prawdopodobieństwa przynależności do klasy) oraz metody służące do głosowania miękkiego i twardego. Estymator przyjmuje macierz przykładów uczących oraz wektor etykiet do każdego z nich. Następnie, dla każdego modelu bazowego losowana jest próba bootstrapowa i w zależności od flagi rand_features losowany jest podzbiór cech równy pierwiastkowi kwadratowemu z liczby dostępnych atrybutów, przy

czym zapisywane są indeksy. Po sklonowaniu obiektu bazowego jest on trenowany i zapisywany jako gotowy model bazowy. Poszczególne predyktory w zależności od rodzaju używają swoich metod głosowania: predict używa głosowania twardego a predict_proba głosowania miękkiego. Dla głosowania twardego dla każdego przykładu wyznaczany jest licznik w postaci słownika oraz tupel zawierający nazwę najbardziej popieranej klasy wraz z liczbą głosów. Przechodząc w pętli przez wszystkie wytrenowane modele bazowe pobierane są predykcje. Jeżeli dokonano randomizacji cech dany przykład jest wycinany do atrybutów zastosowanych w procesie uczenia. Potem licznik dla danej predykcji jest inkrementowany a następnie porównywany z najbardziej popieraną klasą. Po zakończeniu pętli rezultat głosowania zapisywany jest do listy wynikowej. Proces głosowania miękkiego jest o wiele prostszy. Zamiast zliczać poszczególne klasy dla każdego z przykładów inicjalizowana jest zmienna oznaczająca sumę predykowanych prawdopodobieństw. Na zakończenie głosowania liczba ta jest dzielona przez liczbę modeli bazowych uzyskując w ten sposób średnią arytmetyczną ze wszystkich prawdopodobieństw.

Do implementacji klasy modelu zespołowego napisano także kilka funkcji pomocniczych do wykonywania eksperymentów. Są to funkcje tworzące: macierz pomyłek, krzywą ROC, wypisujące metryki dokładności, precyzji, czułości, f1 oraz do wykonania k-krotnego sprawdzianu krzyżowego z użyciem dowolnej metody podziału wraz z wypisywaniem metryk na każdy etap walidacji z ostatecznymi statystykami dla całego procesu oceny (średnia, odchylenie standardowe, wartość minimalna, wartość maksymalna).

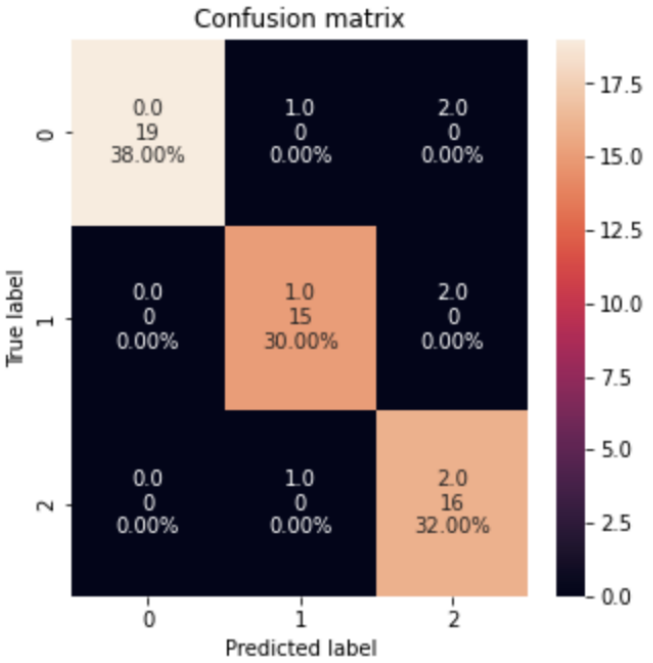
# 7. Testowanie

Przeprowadzono testy na różnych algorytmach klasyfikacji dla zbioru danych iris.

- Zaimportowano potrzebne biblioteki, takie jak sklearn.svm, numpy, pandas, ensemble, itd.
- Wczytano zbiór danych iris za pomocą funkcji datasets.load_iris() i przypisano go do zmiennej iris.
- Zdefiniowano zbiór cech (X) i etykiet (y) na podstawie danych wczytanych ze zbioru iris.
- Podzielono zbiór danych na zbiór treningowy i testowy przy użyciu funkcji train_test_split. Zbiór testowy stanowi 33% danych, a wartość losowa 42 została użyta jako seed dla powtarzalności.
- Stworzono model EnsembleClassifier, który jest zbudowany na klasyfikatorze SVC z włączoną predykcją prawdopodobieństwa. Model ten został wytrenowany na zbiorze treningowym.
- Przeprowadzono predykcję prawdopodobieństwa na zbiorze testowym dla modelu EnsembleClassifier oraz dla modelu BaggingClassifier, który jest oparty na klasyfikatorze SVC. Wyniki tych predykcji zostały przypisane do zmiennych test_proba i bag_proba.
- Obliczono wyniki dla metryki różnicy między prawdopodobieństwami predykcji tych dwóch modeli. Otrzymano następujące wyniki: średnia: 0.0261, odchylenie standardowe: 0.0269, wartość minimalna: 0.0002, wartość maksymalna: 0.1229.
- Dokonano predykcji na zbiorze testowym przy użyciu modelu EnsembleClassifier i obliczono metryki dokładności (Accuracy), precyzji (precision), czułości (recall) oraz

miary F1 (f1-score). Otrzymano wyniki: Accuracy: 1.0, Macro precision: 1.0, Macro recall: 1.0, Macro f1-score: 1.0.

```
Accuracy: 1.0
 Macro precision: 1.0
 Macro recall: 1.0
 Macro f1-score: 1.0
```

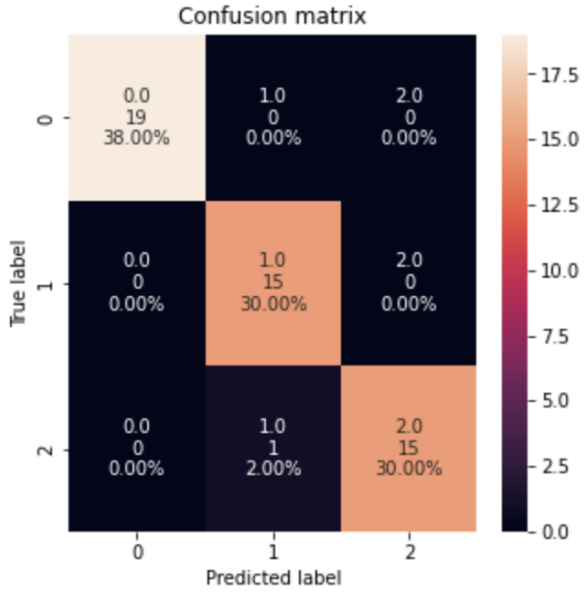



```
7]: {'accuracy': 1.0, 'precision': 1.0, 'recall': 1.0, 'f1-score': 1.0}
```

- Porównano predykcje modelu EnsembleClassifier z rzeczywistymi etykietami ze zbioru testowego. Wartości predykcji zostały przypisane do zmiennej y_test_pred.
- Dokonano predykcji na zbiorze testowym przy użyciu modelu BaggingClassifier opartego na klasyfikatorze SVC i obliczono metryki dokładności (Accuracy), precyzji (precision), czułości (recall) oraz miary F1 (f1-score). Otrzymano wyniki: Accuracy: 0.98, Macro precision: 0.9791666666666666, Macro recall: 0.9791666666666666, Macro f1-score: 0.978494623655914.

```
]:
bag_pred = bag.predict(X_test)
metrics(y_test, bag_pred)
```

```
Accuracy: 0.98
 Macro precision: 0.9791666666666666
 Macro recall: 0.9791666666666666
 Macro f1-score: 0.978494623655914
```

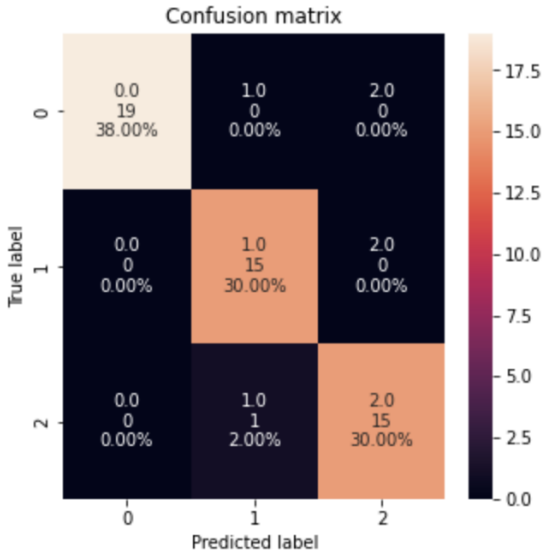



```
]: {'accuracy': 0.98,
    'precision': 0.9791666666666666,
    'recall': 0.9791666666666666,
    'f1-score': 0.978494623655914}
```

- Dokonano predykcji na zbiorze testowym przy użyciu klasyfikatora RandomForestClassifier z 10 estymatorami i obliczono metryki dokładności (Accuracy), precyzji (precision), czułości (recall) oraz miary F1 (f1-score). Otrzymano wyniki: Accuracy: 0.98, Macro precision: 0.9791666666666666, Macro recall: 0.9791666666666666

```
forest_pred = forest.predict(X_test)
metrics(y_test, forest_pred)
```

```
Accuracy: 0.98
 Macro precision: 0.9791666666666666
 Macro recall: 0.9791666666666666
 Macro f1-score: 0.978494623655914
```




```
{'accuracy': 0.98,
 'precision': 0.9791666666666666,
 'recall': 0.9791666666666666,
 'f1-score': 0.978494623655914}
```

- Obliczono metryki dokładności, precyzji, czułości i miary F1 dla predykcji modelu BaggingClassifier na zbiorze testowym przy użyciu funkcji metrics.
- Stworzono model RandomForestClassifier z 10 estymatorami i wytrenowano go na zbiorze treningowym. Następnie dokonano predykcji na zbiorze testowym przy użyciu tego modelu i obliczono metryki dokładności, precyzji, czułości i miary F1 za pomocą funkcji metrics.

# 8.Wnioski

Można wyciągnąć następujące wnioski:

- Model zespołowy oparty na baggingu i lasach losowych może znacznie poprawić skuteczność klasyfikacji w porównaniu do pojedynczych modeli. W przypadku zastosowania lasu losowego jako modelu zespołowego, uzyskujemy większą różnorodność poprzez losowe wybieranie atrybutów i próbek, co przyczynia się do lepszej generalizacji i mniejszego ryzyka przeuczenia.
- Głosowanie miękkie, które uwzględnia prawdopodobieństwo przynależności do klas, może dostarczyć bardziej wiarygodnych wyników niż głosowanie twarde, które opiera się tylko na większości głosów. Użycie głosowania miękkiego pozwala uwzględnić pewność modelu w przyporządkowaniu przykładów do klas.

- Istnieje wiele algorytmów klasyfikacji, które można wykorzystać w modelach zespołowych, takich jak regresja logistyczna, drzewa decyzyjne, k-najbliższych sąsiadów (KNN), maszyny wektorów nośnych (SVM), lasy losowe, Naive Bayes itp. Wybór odpowiednich algorytmów zależy od charakterystyki danych i rodzaju problemu klasyfikacji.
- Badania wykorzystały popularne zbiory danych, takie jak Iris, Breast Cancer, Wine i Pima Indians Diabetes. Wyniki eksperymentów na tych zbiorach danych wskazują na skuteczność modelu zespołowego w porównaniu do pojedynczych modeli. Jednak warto również rozważyć zastosowanie innych zbiorów danych, aby przetestować ogólną wydajność modelu zespołowego w różnych kontekstach.
- Implementacja własnej klasy modelu zespołowego w języku Python jest możliwa przy użyciu pakietu Scikit-Learn. Tworzenie takiego modelu wymaga odpowiedniego doboru parametrów, takich jak liczba estymatorów, losowanie cech, metoda głosowania itp.
- Przeprowadzone badania obejmowały ocenę skuteczności modeli zespołowych i pojedynczych modeli za pomocą różnych metryk, takich jak dokładność, precyzja, czułość, miara F1, macierz pomyłek oraz krzywa ROC wraz z AUC. Analiza tych metryk pozwala na ocenę jakości klasyfikacji i porównanie różnych podejść.
- Wnioski dotyczące czasu trenowania i predykcji różnych algorytmów klasyfikacji nie zostały szczegółowo omówione w dostarczonym tekście. Warto jednak zwrócić uwagę na ten aspekt, ponieważ wydajność obliczeniowa może mieć znaczenie przy skalowaniu modelu zespołowego do większych zbiorów danych.

Podsumowując, model zespołowy oparty na lasach losowych i baggingu może dostarczyć lepszej skuteczności klasyfikacji niż pojedyncze modele. Głosowanie miękkie oraz wybór odpowiednich algorytmów klasyfikacji są istotnymi czynnikami w poprawie jakości predykcji. Projekt dostarcza również kompleksowej dokumentacji, która może być użyteczna przy implementacji i ocenie innych modeli zespołowych w przyszłości.