Zadanie nr 1 - klasyfikacja wzorców

Analiza danych złożonych z detekcją wyjątków

Karol Kazusek - 254189, Sebastian Zych - 254264

14.10.2024

1 Cel zadania

Zadanie polegało na przeprowadzeniu analiz porównawczych klasyfikacji
wzorców (własnych, wyjątkowych) w strumieniach danych. Do analiz należało wybrać tematykę medycyny lub rozpoznawania faz ruchu aktywności człowieka (systemy HAR)

2 Opis zaproponowanych Klasyfikatorów

Klasyfikatory to algorytmy stosowane w *uczeniu maszynowym*, których celem jest przypisanie danych wejściowych do jednej z wcześniej zdefiniowanych kategorii (klas). Klasyfikacja składa się z dwóch głównych etapów: **trenowania** i **predykcji**.

- **Trenowanie** polega na dostarczeniu algorytmowi zbioru danych treningowych, na podstawie którego algorytm uczy się rozpoznawać wzorce i różnicować między klasami.
- **Predykcja** to proces, w którym przetrenowany model jest wykorzystywany do klasyfikowania nowych, wcześniej niewidzianych danych.

Skuteczność klasyfikatorów ocenia się przy użyciu różnych miar, takich jak:

- **Dokładność** (accuracy) — odsetek poprawnie sklasyfikowanych przypadków,
- **Precyzja** (precision) — odsetek prawdziwych pozytywnych wyników spośród wszystkich przykładów sklasyfikowanych jako pozytywne,
- **Czułość** (recall) — odsetek prawdziwie pozytywnych wyników spośród wszystkich pozytywnych przypadków w rzeczywistości,
- **Specyficzność** (specificity) — odsetek prawdziwych negatywnych wyników spośród wszystkich przypadków rzeczywiście negatywnych. Mierzy zdolność klasyfikatora do prawidłowego rozpoznawania negatywnych przykładów, co jest szczególnie istotne w przypadku, gdy negatywna klasa jest dominująca.

Klasyfikatory znajdują zastosowanie w wielu dziedzinach, takich jak medycyna, rozpoznawanie obrazów, analiza tekstu oraz inne zadania związane z przetwarzaniem danych.

2.1 Metody klasyfikacyjne

2.1.1 K-nn

Metoda **K-Najbliższych Sąsiadów (K-NN)** to jedna z najprostszych technik klasyfikacji. Działa na zasadzie porównywania nowego punktu danych z istniejącymi przykładami treningowymi. Algorytm wybiera K najbliższych sąsiadów (na podstawie odległości euklidesowej lub innych miar), a następnie przypisuje nowy punkt do klasy, która występuje najczęściej wśród tych sąsiadów.

- `n_neighbors`: Liczba sąsiadów (K) do uwzględnienia przy klasyfikacji.
- `weights`: Sposób ważenia sąsiadów. Możliwe wartości to `'uniform'` (wszyscy sąsiedzi mają taką samą wagę) i `'distance'` (bliżsi sąsiedzi mają większy wpływ).
- `metric`: Miara odległości używana do porównania punktów,która w zależności od parametru może reprezentować odległość euklidesową lub Manhattan.

2.1.2 Klasyfikator baysowski

Klasyfikacja Bayesowska opiera się na *twierdzeniu Bayesa*, które opisuje zależność między prawdopodobieństwem wystąpienia klasy a dostarczonymi danymi. W najprostszym przypadku, **naiwny klasyfikator Bayesa** zakłada, że cechy są niezależne od siebie. Model wylicza prawdopodobieństwo każdej klasy na podstawie danych wejściowych, a następnie przypisuje nowy przykład do klasy o największym prawdopodobieństwie.

- `var_smoothing`: Parametr ten reguluje wielkość wygładzania (dodanie małej stałej do wariancji, aby uniknąć dzielenia przez zero). (Wygładzanie Laplace).

2.1.3 Drzewa decyzyjne

Drzewa decyzyjne to metoda klasyfikacji oparta na strukturze drzewa, gdzie każdy węzeł reprezentuje decyzję na podstawie jednej cechy, a każda gałąź odpowiada wynikom tej decyzji. Proces ten powtarza się rekurencyjnie, aż do osiągnięcia końcowych węzłów liściowych, które reprezentują klasy.

- `criterion`: Funkcja oceny podziału. Możliwe wartości to `'gini'` (współczynnik Giniego) i `'entropy'` (entropia).
- `max_depth`: Maksymalna głębokość drzewa. Ograniczenie głębokości może zapobiec nadmiernemu dopasowaniu (overfitting).
- `min_samples_split`: Minimalna liczba próbek potrzebna do podziału w węźle.
- `min_samples_leaf`: Minimalna liczba próbek w liściu. Pozwala na kontrolę wielkości końcowych węzłów.
- `max_features`: Maksymalna liczba cech brana pod uwagę przy każdym podziale. Może to być liczba całkowita, wartość zmiennoprzecinkowa lub `'auto'`, `'sqrt'`, `'log2'`.

2.1.4 Lasy losowe

Lasy losowe to technika zespołowa oparta na wielu drzewach decyzyjnych. Każde drzewo jest trenowane na losowym podzbiorze danych oraz cech. Wyniki klasyfikacji uzyskuje się poprzez głosowanie większościowe wśród wszystkich drzew. Lasy losowe zmniejszają problem nadmiernego dopasowania, który jest powszechny w pojedynczych drzewach decyzyjnych.

- `n_estimators`: Liczba drzew decyzyjnych w lesie. Większa liczba drzew zwiększa stabilność predykcji.
- `criterion`: Kryterium oceny podziału, takie jak `'gini'` lub `'entropy'`, podobnie jak w przypadku drzew decyzyjnych.
- `max_features`: Maksymalna liczba cech brana pod uwagę przy każdym podziale.
- `bootstrap`: Jeśli `True`, to próbki są losowane z zamianą (bootstrap). Jeśli `False`, nie jest stosowana zamiana.
- `max_depth`, `min_samples_split`, `min_samples_leaf`: Parametry te działają podobnie jak w przypadku pojedynczych drzew decyzyjnych.

2.1.5 SVM

Maszyny wektorów nośnych (SVM) to metoda klasyfikacji, która stara się znaleźć optymalną hiperpłaszczyznę, która maksymalnie rozdziela dane pomiędzy dwie klasy. SVM może działać zarówno liniowo, jak i nieliniowo, dzięki zastosowaniu tzw. *jąder* (kernels), które przekształcają dane na wyższe wymiary, aby umożliwić rozdzielenie nieliniowych danych.

- `C`: Parametr regularizacji. Wyższa wartość C sprawia, że model bardziej dopasowuje się do danych treningowych, ale może prowadzić do nadmiernego dopasowania.
- `kernel`: Funkcja jądrowa do przekształcania danych. Możliwe wartości to `'linear'`, `'poly'` (wielomianowa), `'rbf'` (jądro radialne) i `'sigmoid'`.
- `gamma`: Parametr jądra, który kontroluje zakres wpływu pojedynczego przykładu treningowego. Może być ustawiony na `'scale'` lub `'auto'`.
- `probability`: Jeśli `True`, to model będzie zwracał prawdopodobieństwa klas, co wymaga dodatkowego obliczenia.

2.1.6 Perceptron Wielowarstwowy (MLP)

Perceptron wielowarstwowy (MLP) to algorytm uczenia głębokiego bazujący na sztucznych sieciach neuronowych. To jedna z najprostszych form sieci neuronowych, będąca modelem uczenia nadzorowanego. Jego struktura składa się z trzech głównych warstw: warstwy wejściowej, jednej lub więcej warstw ukrytych oraz warstwy wyjściowej. Każda warstwa składa się z neuronów, które są podstawowymi jednostkami przetwarzającymi dane.

- `hidden_layer_sizes`: Liczba neuronów w każdej warstwie ukrytej. Można podać jedną wartość (liczba neuronów w jednej warstwie) lub krotkę definiującą liczbę neuronów w kilku warstwach. Domyślnie: (100,).

- `activation`: Funkcja aktywacji dla neuronów. Możliwe wartości to `'identity'`, `'logistic'`, `'tanh'` i `'relu'`.
- `solver`: Algorytm używany do optymalizacji. Możliwe wartości to `'lbfgs'` (optymalizacja metodą quasi-Newtona), `'sgd'` (stochastyczny gradient prosty) i `'adam'` (optymalizacja adaptacyjna).
- `alpha`: Parametr regularyzacji L_2, który zapobiega nadmiernemu dopasowaniu.
- `learning_rate`: Szybkość uczenia. Możliwe wartości to `'constant'`, `'invscaling'` oraz `'adaptive'`.
- `max_iter`: Maksymalna liczba iteracji podczas treningu.

3 Charakterystyka wybranych do danych

MIT-BIH Arrhythmia Database to szeroko stosowany zestaw danych wykorzystywany w badaniach dotyczących analizy sygnałów elektrokardiograficznych (EKG) oraz automatycznej klasyfikacji arytmii. Zbiór ten zawiera 48 zapisów sygnałów EKG zarejestrowanych u 47 pacjentów, w tym zarówno osób z różnymi typami arytmii, jak i zdrowych. Każdy zapis obejmuje około 30 minut sygnału EKG, który został pobrany w częstotliwości 125 Hz.

Dane zostały ręcznie oznakowane przez kardiologów, co pozwala na identyfikację różnych typów arytmii, takich jak np. skurcze dodatkowe, migotanie przedsionków czy blokady serca. MIT-BIH Arrhythmia Database jest często wykorzystywany w algorytmach sztucznej inteligencji do trenowania systemów do automatycznej diagnozy arytmii.
Kroki używane do ekstrakcji uderzeń z sygnału EKG według autorów [?] były następujące:

1. Podział ciągłego sygnału EKG na okna 10s i wybór jednego okna 10s z sygnału EKG.
2. Normalizacja wartości amplitudy do zakresu od zera do jeden.
3. Znalezienie zbioru wszystkich lokalnych maksimów na podstawie zerokrosów pierwszej pochodnej.
4. Znalezienie zbioru kandydatów na szczyty R EKG poprzez zastosowanie progu 0.9 na znormalizowanej wartości lokalnych maksimów.
5. Znalezienie mediany interwałów czasowych R-R jako nominalnego okresu bicia serca dla danego okna (T).
6. Dla każdego szczytu R wybór części sygnału o długości równej 1.2T.
7. Uzupełnienie każdej wybranej części zerami, aby jej długość była równa zdefiniowanej stałej długości.

Liczba próbek: 21900
Liczba kategorii: 5
Częstotliwość próbkowania: 125Hz

Nazwa klasy	etykieta	liczebność
Normalne uderzenie	0	1658
Przedwczesne uderzenie nadkomorowe	1	556
Przedwczesny skurcz komorowy	2	1448
Fuzja skurczu komorowego i normalnego uderzenia	3	162
Niezdyscyplinowane uderzenie	4	1608

4 Eksperymenty i wyniki

4.1 Eksperyment nr 1

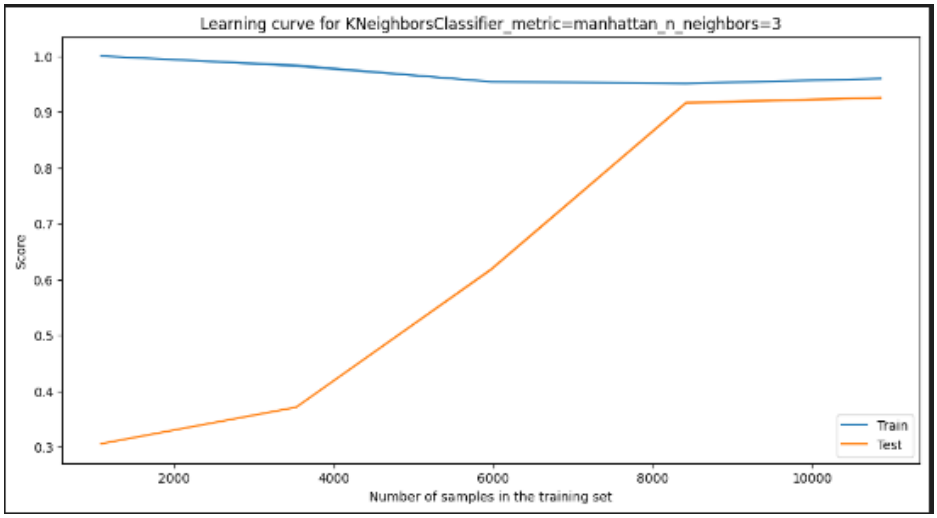
4.1.1 Założenia

Wykonanie porównania dla minimum dwóch klasyfikatorów. Wybrane klasyfikatory to K-nn oraz RF. Dla uproszczenia i oszczedzenia czasu następne graficzne oceny klasyfikatora będą względem najlepszego z Tabeli 1 i 2 odpowiednio dla K-nn i RF.

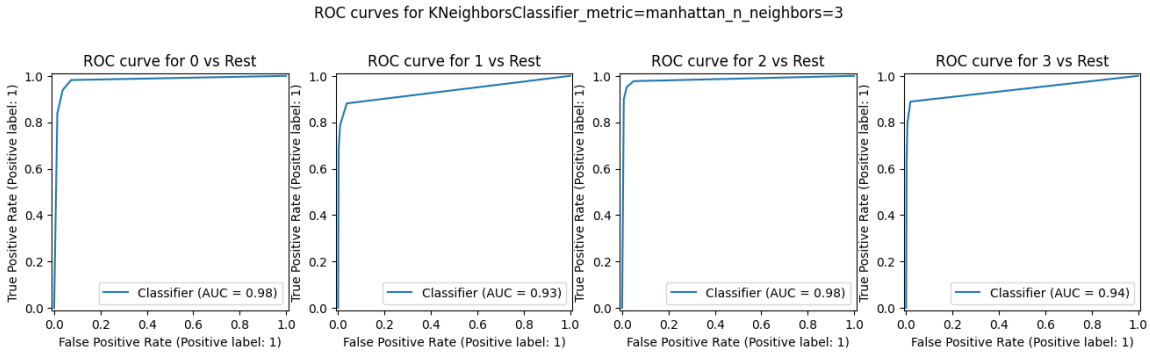
4.1.2 Rezultat K-nn

Classifier	Accuracy	Precision	Sensitivity	Specificity
KNeighborsClassifier_metric=e_n=3	0.929492	0.900287	0.884546	0.981032
KNeighborsClassifier_metric=e_n=5	0.929308	0.902272	0.881949	0.980992
KNeighborsClassifier_metric=e_n=7	0.924521	0.899514	0.876439	0.979617
KNeighborsClassifier_metric=m_n=3	0.937960	0.911232	0.893256	0.983386
KNeighborsClassifier_metric=m_n=5	0.934094	0.904677	0.881621	0.982366
KNeighborsClassifier_metric=m_n=7	0.929124	0.901620	0.876057	0.980939

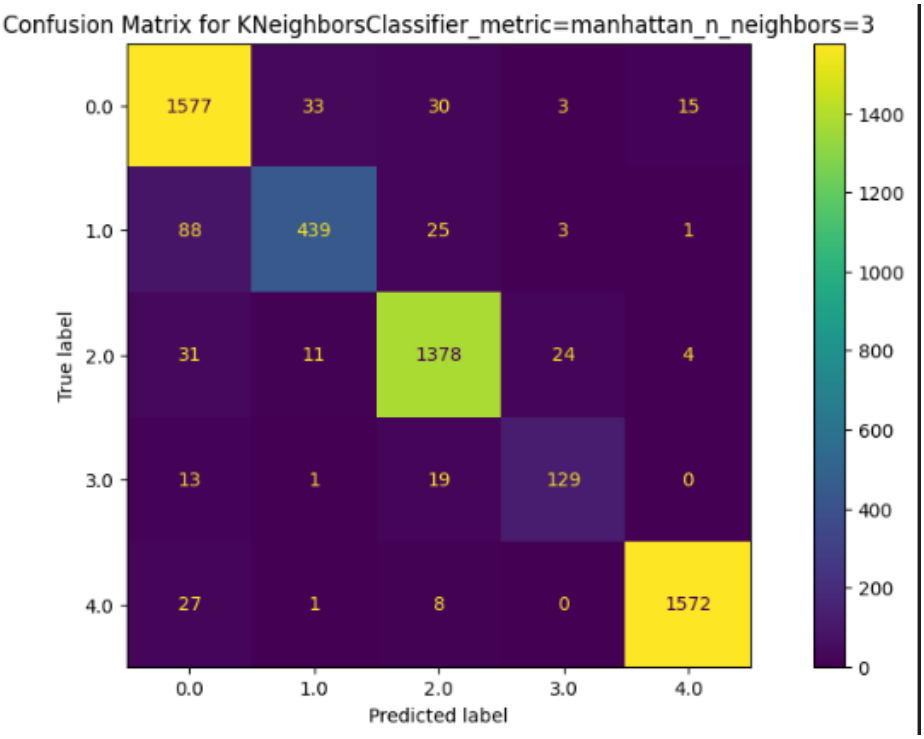
Tabela 1: Statystki K-nn z różnymi metrykami dystansu i liczbą sąsiadów



Rysunek 1: Krzywa uczenia dla K-nn



Rysunek 2: Krzywe AUROC dla K-nn

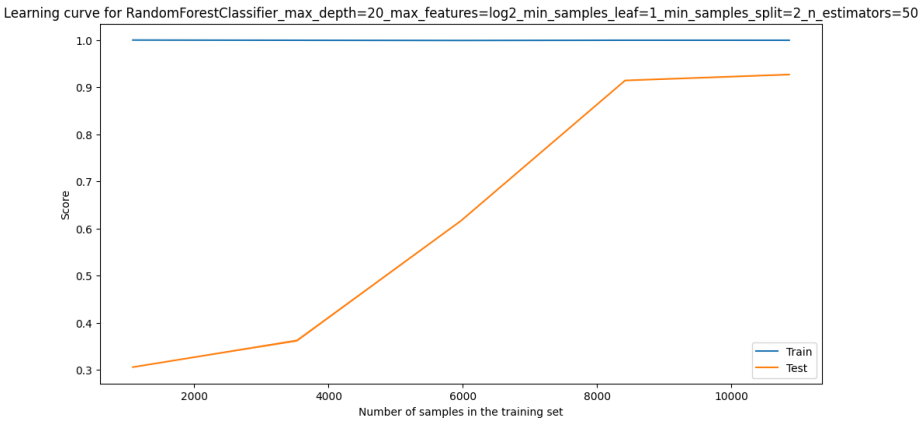


Rysunek 3: Macierz pomyłek dla K-nn

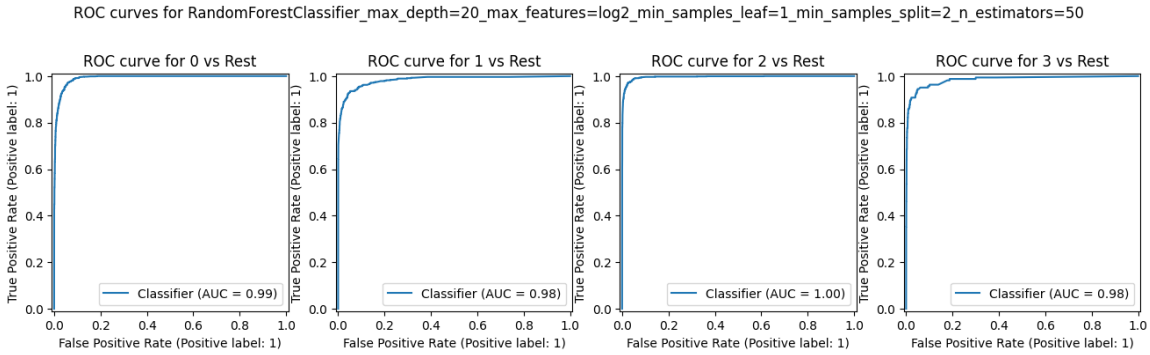
4.1.3 Rezultat RF

Classifier	Accuracy	Precision	Sensitivity	Specificity
RF_d=10_f=log2_l=1_s=2_n=50	0.901694	0.903927	0.810468	0.972396
RF_d=10_f=log2_l=1_s=10_n=50	0.899485	0.905113	0.806056	0.971709
RF_d=10_f=log2_l=5_s=2_n=50	0.903903	0.915494	0.813242	0.972906
RF_d=10_f=log2_l=5_s=10_n=50	0.899853	0.905611	0.805944	0.971839
RF_d=20_f=log2_l=1_s=2_n=50	0.930781	0.927284	0.868739	0.980685
RF_d=20_f=log2_l=1_s=10_n=50	0.929492	0.928302	0.857599	0.980289
RF_d=20_f=log2_l=5_s=2_n=50	0.925074	0.925280	0.851705	0.979075
RF_d=20_f=log2_l=5_s=10_n=50	0.927651	0.925083	0.857425	0.979810

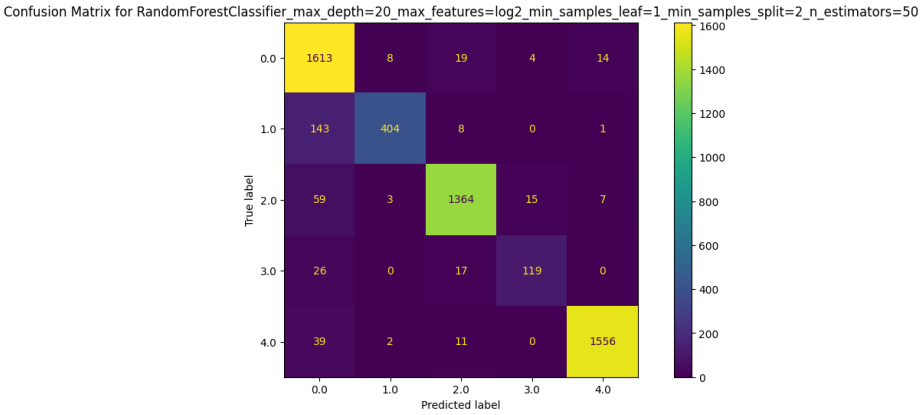
Tabela 2: Statystki RF z różnymi metrykami



Rysunek 4: Krzywa uczenia dla RF



Rysunek 5: Krzywe AUROC dla RF



Rysunek 6: Macierz pomyłek dla RF

4.2 Eksperyment nr 2

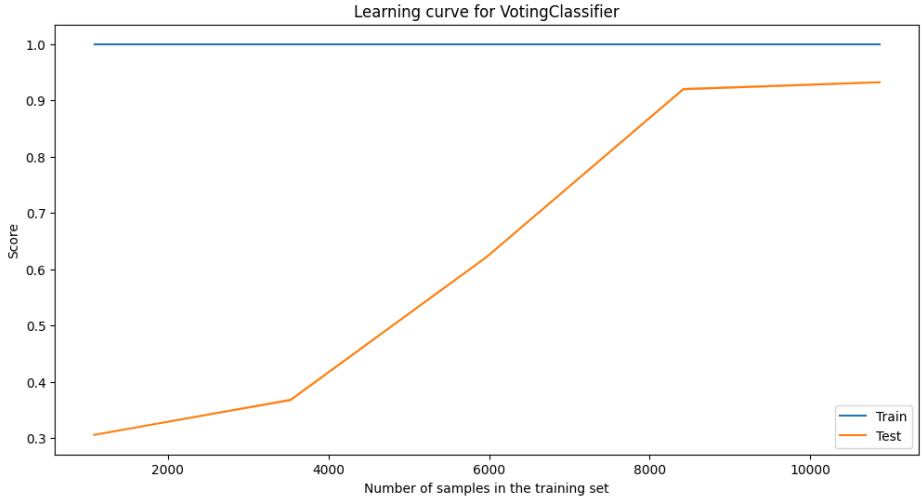
Zaproponowanie 3 klasyfikatorów zespołowych i porównianie ich efektywności. Wybrane klasyfikatory zespołowe to:.

- K-nn + RF + NB
- K-nn + RF + DT
- DT + SVC + MLP

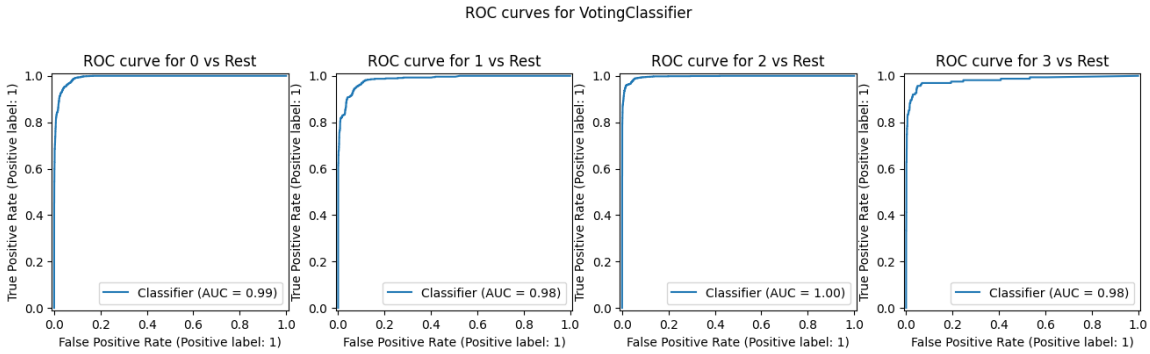
4.2.1 Rezultat K-nn + RF + DT

Classifier	Accuracy	Precision	Sensitivity	Specificity
VotingClassifier	0.938144	0.912708	0.892787	0.983395

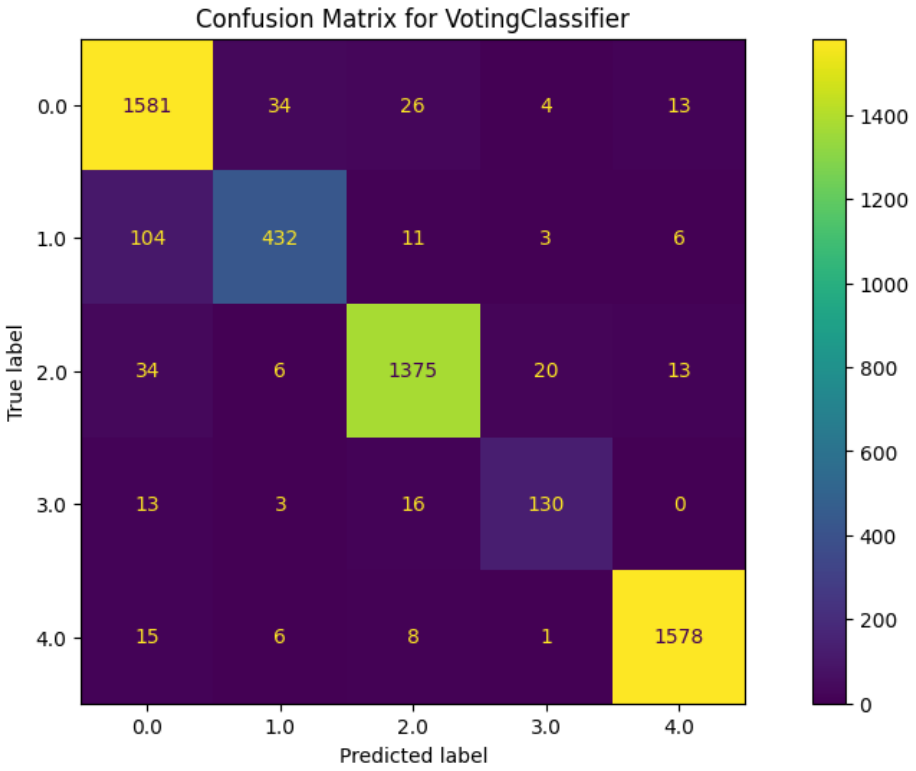
Tabela 3: Statystki K-nn + RF + DT



Rysunek 7: Krzywa uczenia dla KNN+DT+RF



Rysunek 8: Krzywe AUROC dla KNN+DT+RF

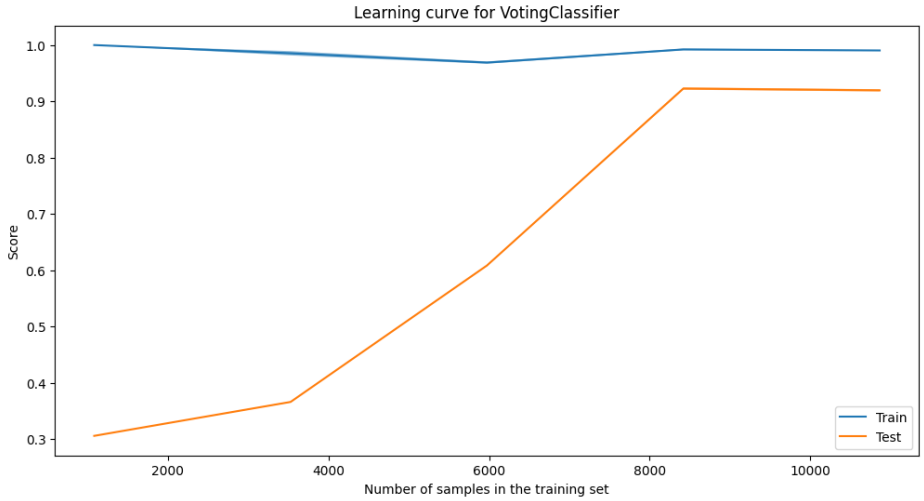


Rysunek 9: Macierz pomyłek dla KNN+DT+RF

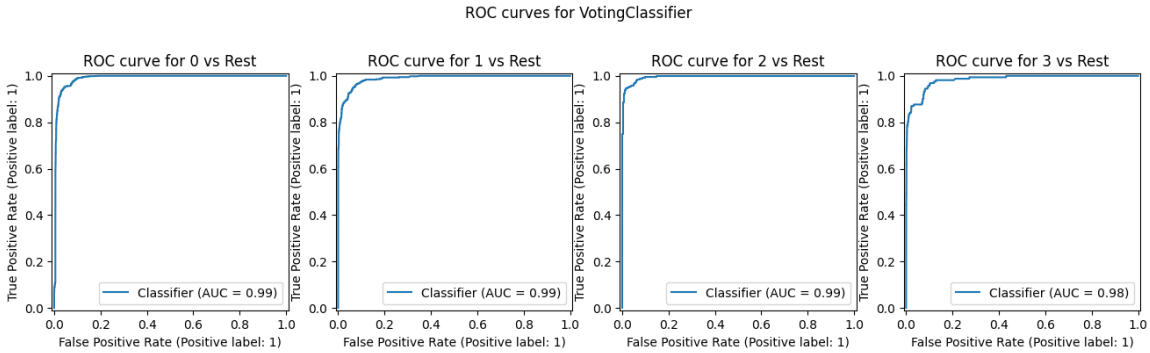
4.2.2 Rezultat K-nn + RF + NB

Classifier	Accuracy	Precision	Sensitivity	Specificity
VotingClassifier	0.928019	0.888637	0.890537	0.980738

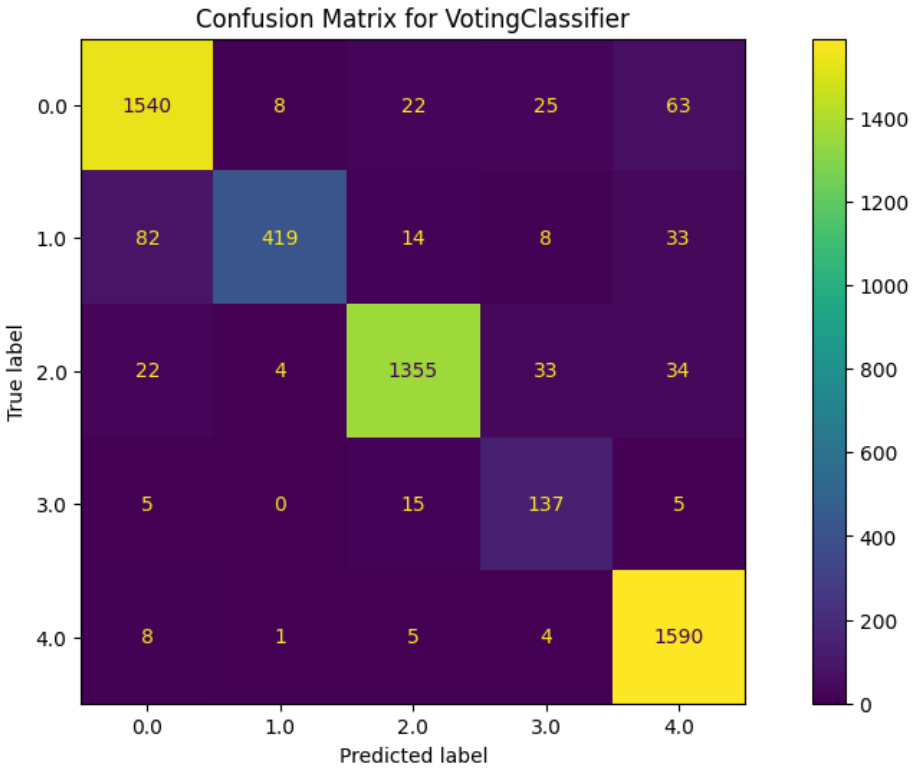
Tabela 4: Statystki dla KNN+RF+NB



Rysunek 10: Krzywa uczenia dla KNN+RF+NB



Rysunek 11: Krzywe AUROC dla KNN+RF+NB

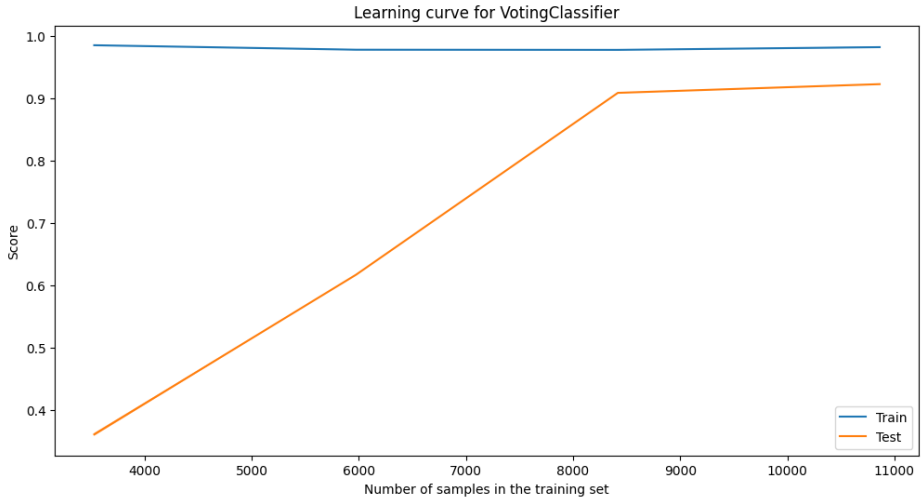


Rysunek 12: Macierz pomyłek dla KNN+RF+NB

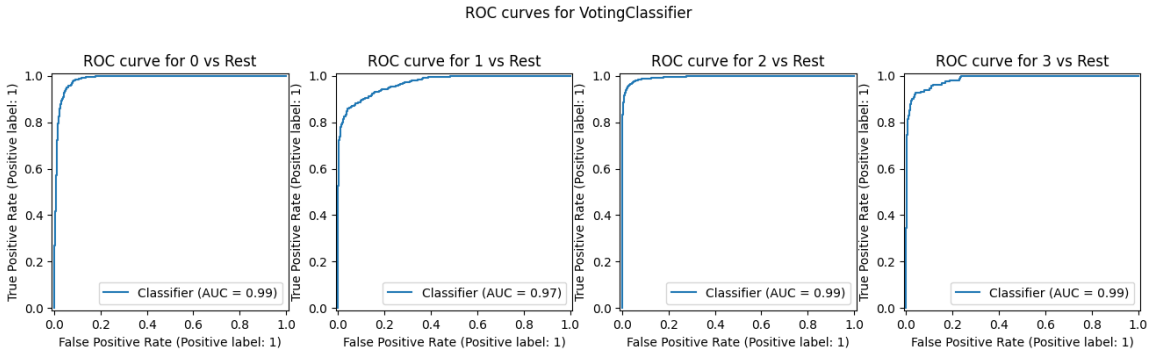
4.2.3 Rezultat DT + SVC + MLP

Classifier	Accuracy	Precision	Sensitivity	Specificity
VotingClassifier	0.927651	0.895403	0.882166	0.980752

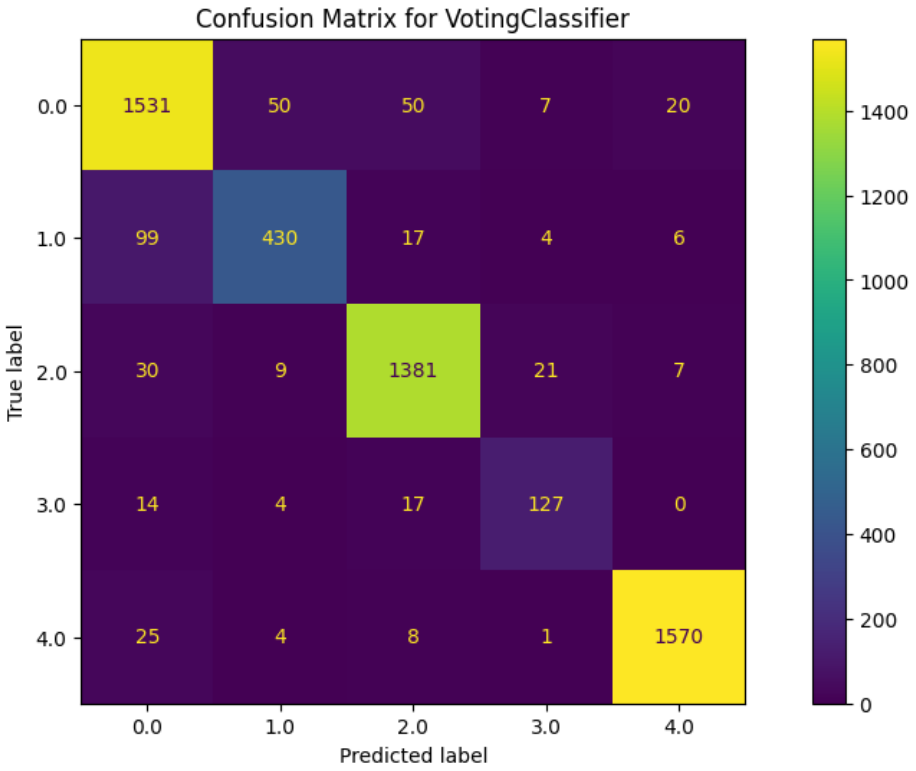
Tabela 5: Statystki dla DT + SVC + MLP



Rysunek 13: Krzywa uczenia dla DT+SVC+MLP



Rysunek 14: Krzywe AUROC dla DT+SVC+MLP



Rysunek 15: Macierz pomyłek dla DT+SVC+MLP

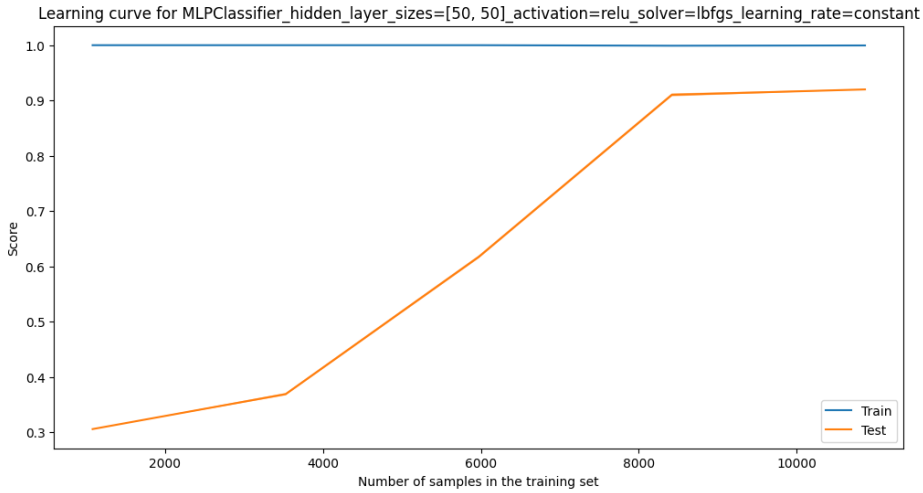
4.3 Eksperyment nr 3

Utworzenie sieci neurownowej i wytrenowania jej w celu klasyfikacji wzorców. Wybraliśmy sieci MLP przez jej prostą implementację, alternatywą mogła być sieci MADALINE.

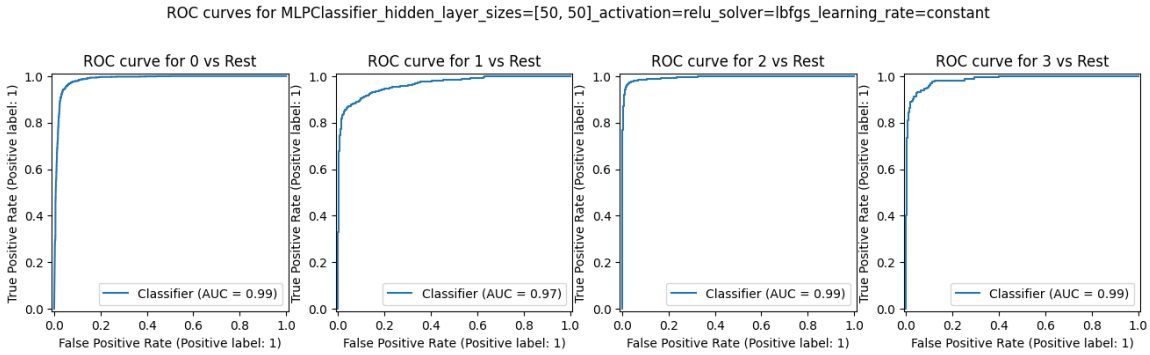
4.3.1 Rezultat MLP

Classifier	Accuracy	Precision	Sensitivity	Specificity
MLP_h_l_s=[20, 20]_a=logistic_l_rate=constant	0.916237	0.892466	0.865486	0.977383
MLP_h_l_s=[20, 20]_a=relu_l_rate=constant	0.923417	0.882847	0.871027	0.979630
MLP_h_l_s=[20, 20]_a=tanh_l_rate=constant	0.921944	0.887618	0.879874	0.979344
MLP_h_l_s=[50, 50]_a=logistic_l_rate=constant	0.932806	0.890531	0.896438	0.982492
MLP_h_l_s=[50, 50]_a=relu_l_rate=constant	0.933542	0.888540	0.895065	0.982791
MLP_h_l_s=[50, 50]_a=tanh_l_rate=constant	0.932990	0.893440	0.890982	0.982414

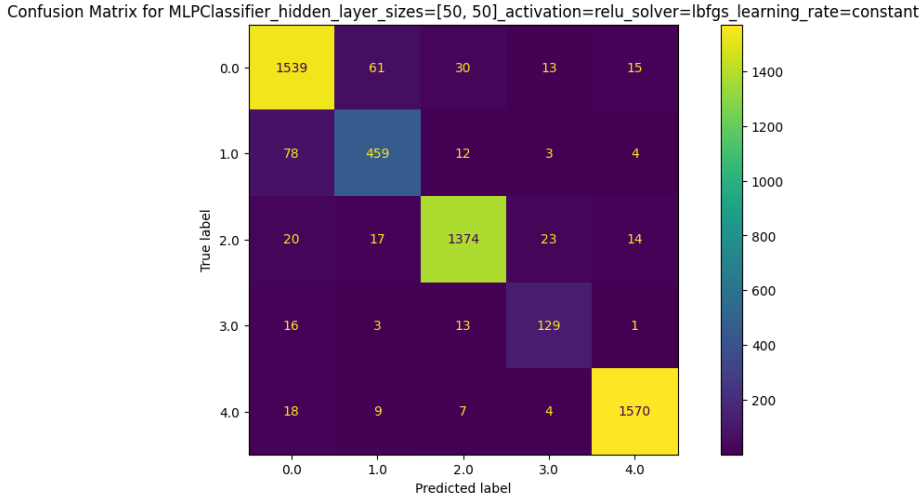
Tabela 6: Statystki dla Klasyfikatora MLP z róznymi rozmiaramy warstwy ukrytych (2), funkcji aktwacji



Rysunek 16: Krzywa uczenia dla MLP



Rysunek 17: Krzywe AUROC dla MLP



Rysunek 18: Macierz pomyłek dla MLP

5 Wnioski

Wnioski z przeprowadzonych eksperymentów dowodzą, że

- Efektywność poszczególnych klasyfikatorów w dużej mierze zależy od poprawnego wyboru jego parametrów.
- W wiekszości przypadków klasyfikatory zespołowe są lepszę od pojedynczych klasyfikatorów, ponieważ poszczególne klasyfikatory posiadają zróżnicowane silne jak i słabe strony. Przez głosowanie, słabe strony i przez to błędy pojedynczych klasyfikatorów są skorygowane przez głosy innych klasyfikatorów. Dzieki czemu zyskaliśmy najlepszy klasyfikator zespołowy K-nn + RF + NB.
- Ciekawym elementem do zbadania mogło być prób stworzenia zespołowego klasyfikatora składającego się z pojedynczego klasyfikatora z różnymi parametrami.
- Sieci Neuronowa MLP uzyskała porównywalny wynik do innych klasyfikatorów, lecz przez wysokie wymagania wydajnościowe nie byliśmy w stanie przetrenować sieci z wszystkim możliwimy parametrami, najważniejszymi z nich jest ilość warstw i neuronów, ilość epok oraz stała uczenia.
- Zbiór danych był zbalanosoway zwględem liczebności klas, jednyną klasą o bardzo małej ilości była Fuzja skurcz komorowego i normalnego uderzenia, przez to uzyskiwała największy błąd klasyfikatora.