Projekt

Algorytmy uczenia maszynowego

Założenia projektowe

Klasyfikacja gestów dłoni

Skład grupy:
Leonard Feszczuk, 249027

Termin: Pn Tp 17:05

Prowadzący:
Mgr inż. Marcin Ochman

20 czerwca 2022

Spis treści

1 Opis projektu **2**
1.1 Założenia projektowe 2
1.2 Zakres prac 2

2 Harmonogram pracy **2**
2.1 Opis kamieni milowych 2

3 Dane pomiarowe **3**

4 Algorytmy **4**
4.1 Model K najbliższych sąsiadów 4
4.2 Model MLP 5
4.3 Model SVC 5

5 Wykresy i miary **6**

1 Opis projektu

Projekt dotyczy zbudowania modelu uczenia maszynowego uczącego się na podstawie pomiarów z rękawicy sensorycznej, powstałej na potrzeby pracy inżynierskiej, tak abyśmy mogli rozpoznawać gesty. Dane z rękawicy przesyłane są do komputera za pomocą USB i rejestrowane w programie stworzonym w środowisku Qt. Pojedynczy gest składa się z 16 pomiarów z czujników ugięcia zapisanych na 10 bitach, wyrażony w omach.

1.1 Założenia projektowe

Zebranie danych do zbioru uczącego oraz testowego. Wykorzystanie bibliotek pythona scikit-learn, pandas do opracowania modelu.Wykorzystanie repozytorium git do kontroli wersji programu. Wizualizacja otrzymanych wyników za pomocą biblioteki matplotlib. Program składa się z dwóch trybów tryb treningowy oraz tryb normalny. Określenie około 5 użytych gestów. Określenie cech oraz klasyfikacja gestów. Opracowanie trzech algorytmów uczenia maszynowego.

1.2 Zakres prac

- Opracowanie danych do zbioru uczącego i testowego
- Opracowanie algorytmów uczenia maszynowego.
- Implementacja oraz łączenie wybranych algorytmów
- Ocena wydajności zastosowanych algorytmów.
- Wizualizacja otrzymanych wyników.

2 Harmonogram pracy

2.1 Opis kamieni milowych

28.03 Zakończenie prac przygotowawczych, środowisko i narzędzia gotowe, założone repozytorium git.

25.04 Koniec opracowywania danych. Dane są zapisane oraz zebrane w zbiory.

09.05 Zakończenie prac nad algorytmami, pierwsze wyniki.

23.05 Zaawansowane prace nad algorytmami, ocena wydajności algorytmów.

06.06 Wizualizacja wyników.

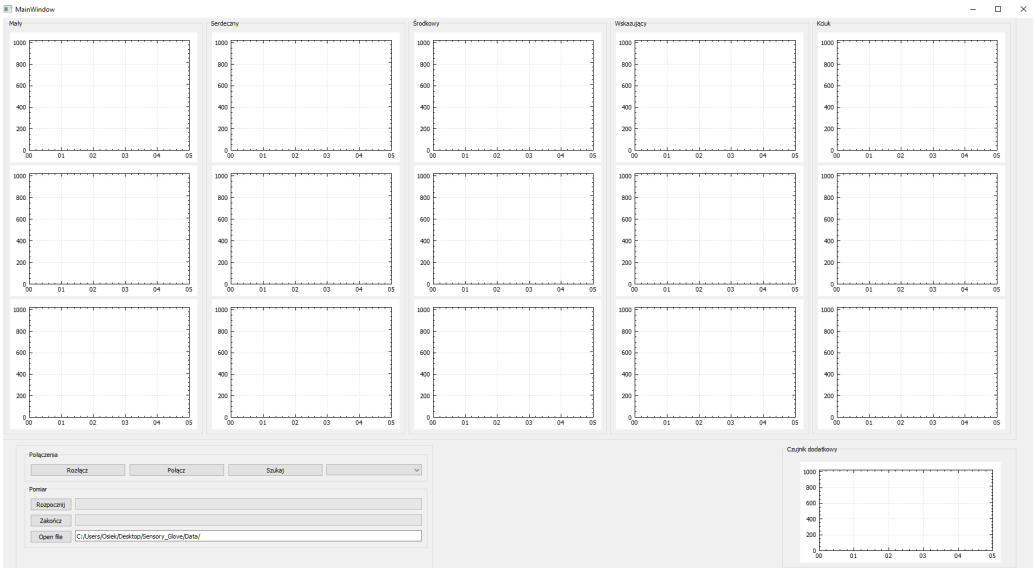
20.06 Zakończone zostają testy programu, projekt jest gotowy do oddania.

3 Dane pomiarowe

Dane użyte w programie zostały pobrane z rękawicy sensorycznej na której zostało zamontowanych 16 czujników ugięcia zmieniających swoją rezystancję w zależności od nacisku. Wartość z każdego czujnika sczytywana jest poprzez wejście ADC płytki Arduino micro. Wartości te są z zakresu od 0 do 1023. Następnie poprzez kabel USB Przesyłane są one do aplikacji na komputerze w QT. Kolejno dane zapisywane są do plików tekstowych skąd pobierane są do wstępnej obróbki a następnie w programie używane są do modeli uczenia maszynowego.

W programie użyto 5 klas/gestów są to:

- ok - Zaciśnięta pięść i kciuk w górę
- peace - Wyprostowane palce wskazujący i środkowy
- bottle - Trzymana w ręku szklana butelka
- flat - Ręka płasko położona na blacie
- pointing - Wyprostowany palec wskazujący



Rysunek 1: Okno aplikacji w Qt

Proces wstępnej obróbki danych przebiega następująco:

1. Dane wczytywane są do data frame pandas
2. Usuwane są trzy kolumny niepotrzebnych danych

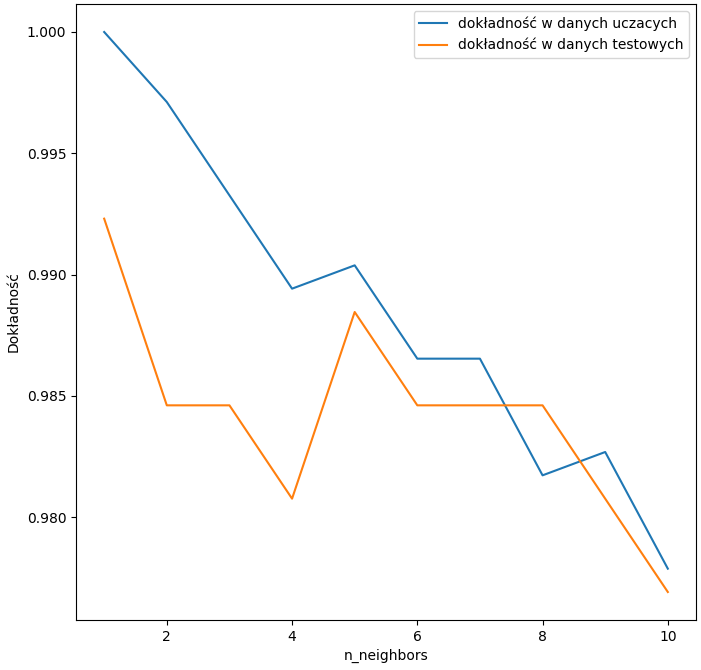
3. Usuwane są niepotrzebne spacje oraz następuje rzutowanie danych do int
4. Następnie dane przeskalowywane są do zakresu od 0 do 1
5. Zapis danych do pliku wynikowego

4 Algorytmy

4.1 Model K najbliższych sąsiadów

Algorytm ten przyporządkowuje nowy punkt danych na podstawie przeważającej ilości najbliższych punktów danych. Najważniejszym modyfikatorem tego algorytmy jest ilość sąsiadów, im większa ich ilość tym model jest prostrzy a dla mniejszych wartości może się okazać zbytnio dopasowany. N wykresie możemy zaobserwować że najlepszym stosunkiem uogólniania oraz wyniku jest liczba 5 sąsiadów.

```
estimator =  KNeighborsClassifier(n_neighbors=5)
```



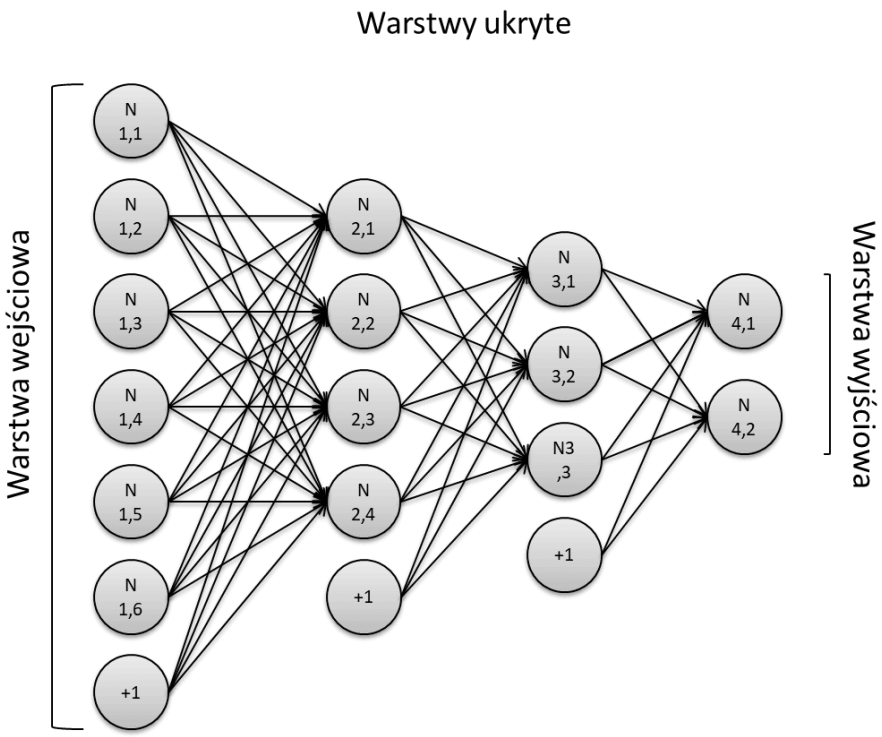
Rysunek 2: Wykres krzywej uczenia KNeighbors w zależności od ilości sasiadów

4.2 Model MLP

Model MLP(ang. Multilayer perceptrons) czyli sieć neuronowa jest to algorytm uczenia maszynowego posiadający pewną ilość warstw wejść i wyjść. Głębokie uczenie jest niezwykle obiecujące w wielu aplikacjach uczenia maszynowego a jego algorytmy są często bardzo dokładnie dostosowywane do konkretnych przypadków użycia. Modele mlp składają się z pewnej ilości warstw ukrytych na która składa się kilka jednostek, dodatkowo można również dostrajać funkcję aktywacji odpowiadającą za dobór wag między perceptronami.

Użyte współczynniki:

```
estimator = MLPClassifier(activation='relu',max_iter=5000, alpha=1, random_state=42,
    hidden_layer_sizes=[10,10])
```

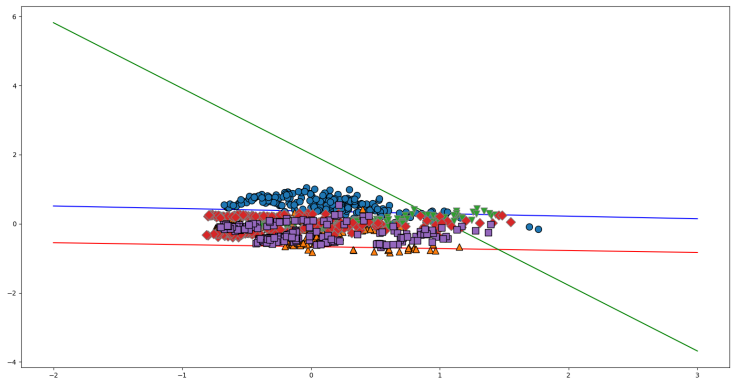


Rysunek 3: Warstwa sieci neuronowej

4.3 Model SVC

SVC to model nadzorowany maszyn wektorów nożnych z jądrem. Jest to model liniowy do klasyfikacji, którego zasadą działania jest podzielenie przestrzeni cech na obszary poprzez odpowiednie wyznaczenie linii podziału klas. Aby zaprezentować granicę decyzyjną między dwoma klasami,

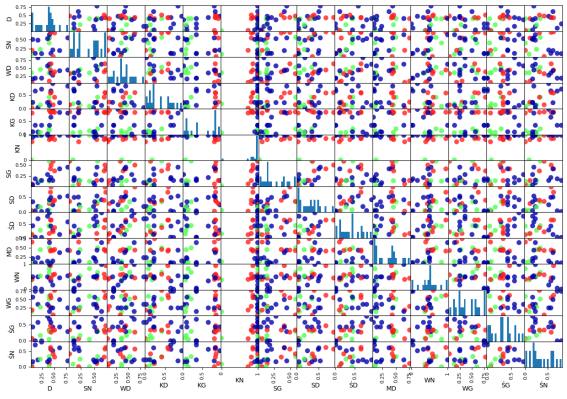
maszyna SVM uczy się, jak ważny jest każdy z punktów danych uczących. Zwykle dla określenia granicy decyzyjnej ma znaczenie tylko podzbiór punktów uczących: tych , które leżą na granicy między klasami. Nazywane są one wektorami nośnymi i to od nich bierze się nazwa maszyny wektorów nośnych.



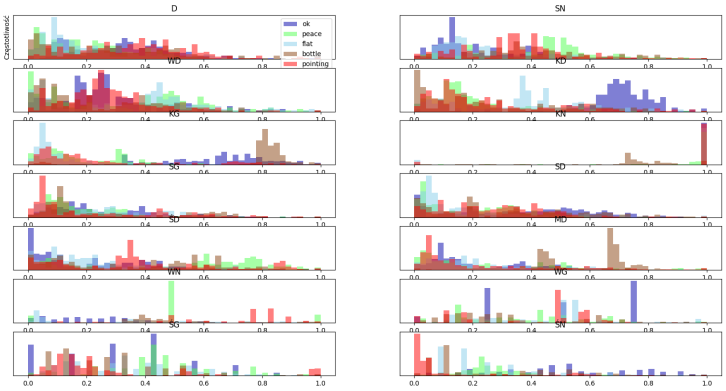
Rysunek 4: Linie podziału maszyny SVC

5 Wykresy i miary

W programie wyrysowano szereg wykresów mających zilustrować dane oraz skuteczność modeli uczenia. Początkowa ilość cech nie pozwalała na jasną interpretację oraz rozpoznanie własności klas. Jak widać rysunek 5 ma zbyt dużą ilość cech aby w ten sposób można było coś wywnioskować. Na ryzunku 6 widzimy histogramy dla poszczególnych cech.

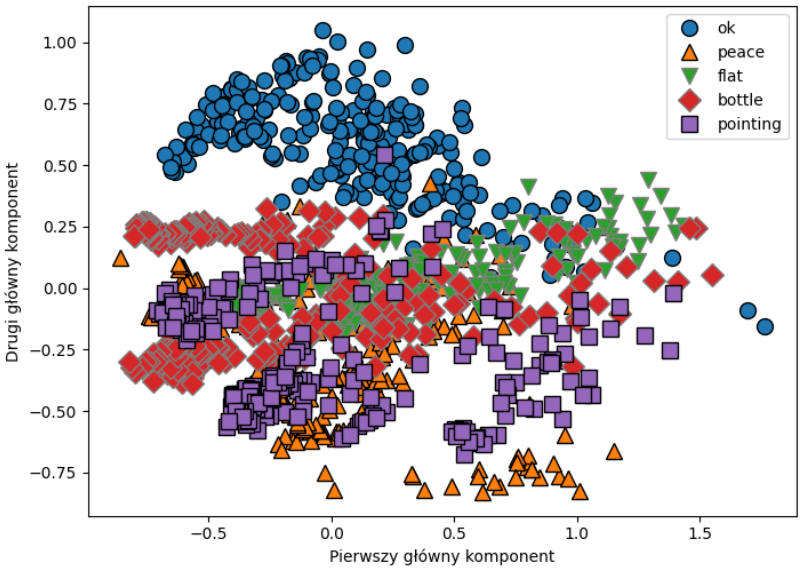


Rysunek 5: Wykresy przestrzeni cech

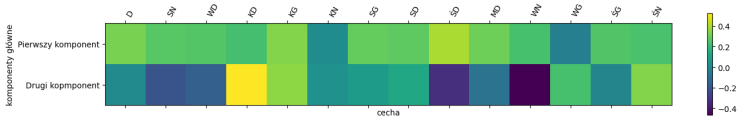


Rysunek 6: Histogramy cech dla poszczególnych klas

Aby zmniejszyć ilość cech dla klasy wykorzystano analizę głównych komponentów PCA(ang. prinicipial components analisys). Co pozwala nam stworzyć nową wybraną ilość cech na podstawie starych. W tym przypadku wybrana ilośc to dwie cechy.
Dzięki temu możemy wyróżnić zauważalne zbiory punktów danych z poszczególnych klas. Możemy również przyjrzeć się bliżej wartością wag dla poszczególnej starej cechy która weszła w skład nowej dzięki mapie cieplnej.

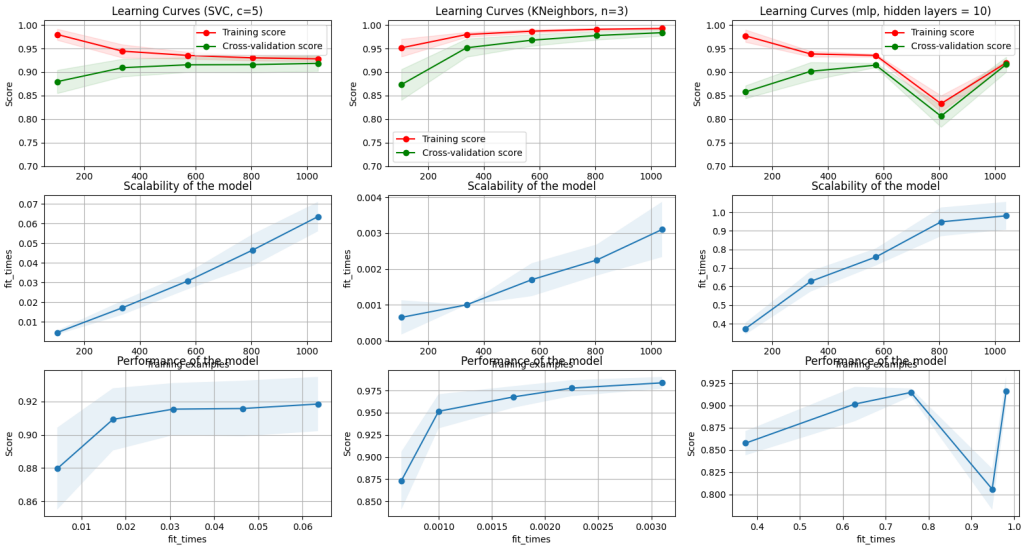


Rysunek 7: Dwie cechy po analizie PCA



Rysunek 8: Mapa cieplna PCA

Najważniejszymi metrykami są krzywe uczenia. Na Rysunku 9 zaprezentowane są wyniki trzech algorytmów w zależności od ilości punktów ich czas w zależności od ilości punktów oraz wynik od czasu. Dzięki temu możemy zauważyć że najszybszy oraz z najlepszymi wynikami jest algorytm K sąsiadów. Oraz że złotym środkiem dla wszystkich algorytmów jest ilość mniej więcej 600 próbek.



Rysunek 9: Krzywe uczenia dla trzech modeli

Literatura

[1] Andreas Muller.Machine learning, Python i data science : wprowadzenie. Gliwice: Helion, 2021.isbn: 9788328374089