Maksymilian Romańczuk

nr albumu GD41582

**Algorytm Kernel SVM**

**Przy użyciu języka Python i biblioteki sklearn i SVM**

Gdańsk 14.12.19

**Wstęp:**

**Cel:**

Kategoryzacja irysów poprzez ich atrybuty metodą nieliniową SVM Kernela.

**Technologie:**

**Scikit-learn**(znane także, jako sklearn) - jest top darmowa biblioteka, którą można wykorzystać przy uczeniu maszynowym stworzona dla języka Python, która pozwala na stworzeniu algorytmów klasyfikacji, regresji, i analizy skupień. Jest tak zaprojektowana żeby interpretery Pythona mogły używać bibliotek numerycznych takich jak NumPy czy SciPy mogły z nim działać. (W moim wypadku wykorzystanie NumPy).

**Support Vector Machine**(znane także, jako SVM) - jest to algorytm linowej klasyfikujący, który ma na celu wyznaczenie hiperpłaszczyzny rozdzielającej maksymalnym marginesem należące do dwóch klas. Najczęściej wykorzystywany przy rozpoznawaniu obrazów

**Kernel SVM**- jw. Ze zmianą w klasyfikacji, która w tym wypadku jest nie liniowa.

**Train\_test\_split**-metoda do podzielenia bazy na grupy trenujące i grupy testujące w tym wypadku jest to 80% procent trenujących i 20% testujących

**Pandas**- biblioteka stworzona dla języka Python, aby analizować i manipulować danymi. Oferuje użytkownikowi, operacje na strukturach danych i manipulowanie tabelami numerycznymi.

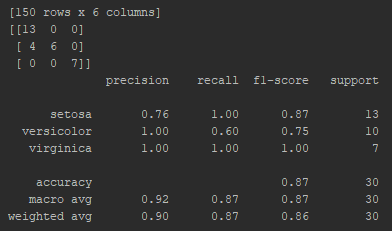
**PyCharm**- IDE dla języka programowania Python firmy JetBrains.

**Baza Danych**- W tym wypadku wykorzystanej bazy danych o nazwie iris.csv składającej się z 6 kolumn i 150 wierszy. Baz danych przechowuje dane na temat 3 gatunków irysów(setosa, versicolor, virginica), i ich atrybutów takich jak długość kielicha, szerokość kielicha, długość płatka, i szerokość płatka

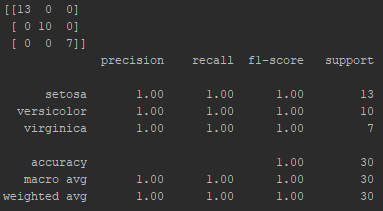
**Kod:**

import pandas as pd #import panads  
  
#wgranie bazy danych iris  
dataset = pd.read\_csv("iris.csv")  
print(dataset)  
  
#usnięcie kolumny typu tekst 'Species'  
X = dataset.drop('Species', axis=1)  
y = dataset['Species']  
  
#funkcja rozdzielająca na grupy testujące 20% i trenujące 80%  
from sklearn.model\_selection import train\_test\_split  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.20)  
  
#wielomianowe wykorzytsanie Kenela  
from sklearn.svm import SVC  
svclassifier = SVC(kernel='poly', degree=8)  
svclassifier.fit(X\_train, y\_train)  
  
#przewidywanie wyniku  
y\_pred = svclassifier.predict(X\_test)  
  
#ocenianie wyniku  
from sklearn.metrics import classification\_report, confusion\_matrix  
  
#wyświetlanie wyniku  
print(confusion\_matrix(y\_test,y\_pred))  
print(classification\_report(y\_test,y\_pred))  
  
#wykorzystanie Kernela metoda Gaussa  
from sklearn.svm import SVC  
svclassifier = SVC(kernel='rbf')  
svclassifier.fit(X\_train, y\_train)  
  
#ocenianie wyniku  
from sklearn.metrics import classification\_report, confusion\_matrix  
  
#wyświetlanie wyniku  
print(confusion\_matrix(y\_test,y\_pred))  
print(classification\_report(y\_test,y\_pred))  
  
#sigmoidalne klasyfikowanie   
from sklearn.svm import SVC  
svclassifier = SVC(kernel='sigmoid')  
svclassifier.fit(X\_train, y\_train)  
  
#ocenianie wyniku  
from sklearn.metrics import classification\_report, confusion\_matrix  
  
#wyświetlanie wyniku  
print(confusion\_matrix(y\_test,y\_pred))  
print(classification\_report(y\_test,y\_pred))

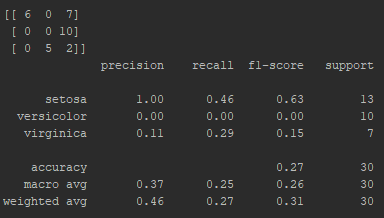
**Wynik klasyfikacji wielomianowej:**



**Wynik klasyfikacji metodą Gaussa:**



**Wynik klasyfikacji sigmoidalnej:**



**Wnioski:**

Podsumowując projekt, przy wykorzystaniu klasyfikacji Gaussa dostajemy najbardziej wiarygodne wyniki gdyż klasyfikuje wszystkie irysy prawidłowo, natomiast klasyfikacja wielomianowa daje nam prawie idealne wyniki z wyjątkiem irysów typu setosa, który wynosi 0.76, za to najgorsze wyniki daje nam metoda sigmoidalna, przy której klasyfikacja nie powinny być brane pod uwagę z powodu ich zaskakujących wyników, które nie dają żadnej dobrej odpowiedzi. Tych algorytmów można używać także do klasyfikacji innych przedmiotów, bądź obrazów, do czego algorytm był głównie stworzony.

Bibliografia:

<https://scikit-learn.org/stable/modules/svm.html>

<https://vincentarelbundock.github.io/Rdatasets/datasets.html?fbclid=IwAR3py416E8AM73f5NVhqAgycl9wtHI5hMaOWIfuvrE4_v9cG9WAoE8kyIyo>

<https://en.wikipedia.org/wiki/Scikit-learn>

<https://pl.wikipedia.org/wiki/PyCharm>

<https://en.wikipedia.org/wiki/Support-vector_machine>

<https://en.wikipedia.org/wiki/Kernel_method#Mathematics:_the_kernel_trick>

<https://en.wikipedia.org/wiki/Sigmoid_function>

<https://en.wikipedia.org/wiki/Gaussian_function>

<https://en.wikipedia.org/wiki/Polynomial_kernel>

<https://towardsdatascience.com/understanding-support-vector-machine-part-2-kernel-trick-mercers-theorem-e1e6848c6c4d>

<http://www.cs.put.poznan.pl/jstefanowski/ml/SVM.pdf>