Київський національний університет імені Тараса Шевченка
Факультет інформаційних технологій

Кафедра програмних систем і технологій

ЗВІТ

з практичної роботи № 3

Тема: «Алгоритми класифікації із застосуванням бібліотек»

Дисципліна «Спеціалізоване програмування автоматизованих систем»

Підготував:
студент гр. ІПЗ-33(1)
Мішак Максим

Перевірила:
Ніколаєнко Анастасія Юріївна

Завдання :
Класифікатор Методу *k*-найближчих сусідів (відстань Хеммінга). Дані з файлу «seeds.csv».

Хід роботи :

Для виконання роботи будемо використовувати наступний перелік бібліотек :

- **pandas**
- **matplotlib**
- **seaborn**
- **sklearn**

Для класифікації будемо використовувати готове рішення з бібліотеки **sklearn** , а саме KNeighborsClassifier . Використання цього методу потребує знання деяких нюансів , до прикладу розділення наданого датасету відбувається наступним чином :

```
X_train, X_test, y_train, y_test =
train_test_split(X, y, test_size=0.2,
random_state=42)
```

цю конструкцію можна записати будь яким зручним способом - але такий запис рахується канонічним . Для інфографіки будемо використовувати елемент бібліотеки **sklearn.metrics classification_report , confusion_matrix.** Це нам потрібно для виведення результатів обробки програмою датасету .

Згідно методичних вказівок , згідно з моїм варіантом я маю датасет з назвою seeds.csv . Структура данних в нього виглядає так :

```
"V1","V2","V3","V4","V5","V6","V7","class"
15.26,14.84,0.871,5.763,3.312,2.221,5.22,1
```

Маємо данні , які репрезентують вектори , деякі з яких суттєво відрізняються значеннями . Для того , щоб ці значення не заважали класифікатору працювати будемо виконувати нормалізацію данних :

```
scaler = StandardScaler()
scaler.fit(X_train)
X_train = scaler.transform(X_train)
X_test = scaler.transform(X_test)
```

```
knn.fit(X_test, y_test)
```

Це дозволяє “форматувати” данні до приблизного одного вигляду та значення .

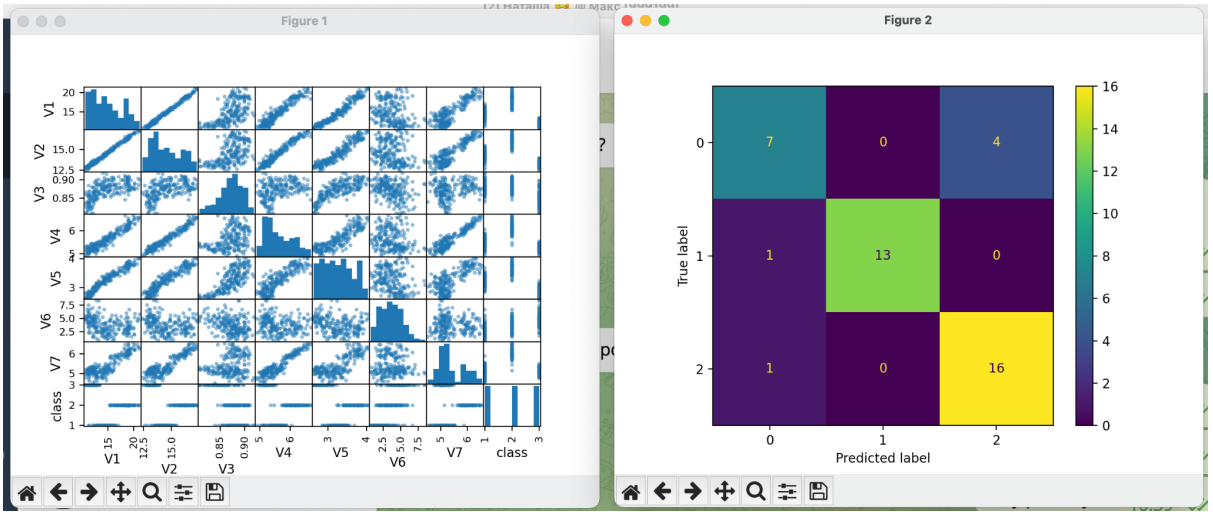
Код застосунку

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sb
from pandas.plotting import scatter_matrix
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import classification_report,
confusion_matrix ,ConfusionMatrixDisplay


#read the data from file
DataSet =
pd.read_csv('/Users/macbook/Desktop/SPAS3/dataset/seeds.csv')
knn = KNeighborsClassifier(n_neighbors=5,
metric='hamming')
#set data as X and y
X = DataSet.iloc[:, :-1].values
y = DataSet.iloc[:, 7].values
#spliting the data fot 2 frames
c
#scale and transform datda
scaler = StandardScaler()
scaler.fit(X_train)
X_train = scaler.transform(X_train)
X_test = scaler.transform(X_test)
```

```
knn.fit(X_test, y_test)
#make a "prediction" for the data
y_pred = knn.predict(X_test)
#print thw defoult data
print(DataSet)
#ptinting of data shapes
print(X_train.shape, X_test.shape, y_train.shape,
y_test.shape)
#printing reports confusion matring and class report
print(classification_report(y_test,y_pred))
cm = confusion_matrix(y_test, y_pred)
print(cm)
#plotting a scatter matrix
scatter_matrix(DataSet)
#plotting the confusion matrix
disp = ConfusionMatrixDisplay(confusion_matrix=cm)
disp.plot()
plt.show()
```

Скріншоти виконання программи




```
Run:  main
/Users/macbook/Desktop/SPAS3/venv/bin/python /Users/macbook/Desktop/SPAS3/venv/main.py
        V1     V2      V3     V4     V5     V6     V7  class
0    15.26  14.84  0.8710  5.763  3.312  2.221  5.220      1
1    14.88  14.57  0.8811  5.554  3.333  1.018  4.956      1
2    14.29  14.09  0.9050  5.291  3.337  2.699  4.825      1
3    13.84  13.94  0.8955  5.324  3.379  2.259  4.805      1
4    16.14  14.99  0.9034  5.658  3.562  1.355  5.175      1
..     ...    ...     ...    ...    ...    ...    ...    ...
205  12.19  13.20  0.8783  5.137  2.981  3.631  4.870      3
206  11.23  12.88  0.8511  5.140  2.795  4.325  5.003      3
207  13.20  13.66  0.8883  5.236  3.232  8.315  5.056      3
208  11.84  13.21  0.8521  5.175  2.836  3.598  5.044      3
209  12.30  13.34  0.8684  5.243  2.974  5.637  5.063      3

[210 rows x 8 columns]
(168, 7) (42, 7) (168,) (42,)
              precision    recall  f1-score   support

           1       0.78      0.64      0.70        11
           2       1.00      0.93      0.96        14
           3       0.80      0.94      0.86        17

    accuracy                           0.86        42
   macro avg       0.86      0.84      0.84        42
weighted avg       0.86      0.86      0.85        42

[[ 7  0  4]
 [ 1 13  0]
 [ 1  0 16]]
```

Version Control | Run | Debug | TODO | Problems | Terminal | Python Packages | Python Console | Services

Висновок :

Було опрацьовано методи класифікації бібліотеки sklearn , а саме метод knn. В результаті можемо побачити , що при теперішніх налаштуваннях

функції пошуку найблищого сусіда середня ефективність є 86% . Така точність була досягнута шляхом емпіричного підбору параметра кількості найблищих сусідів . Для кожного окремого класу це показник варіюється , так як різні класи можуть приблизно співпадати по показникам .