Computer Engineering Department
National University of Technology Islamabad, Pakistan

Introduction to Data Mining

Practice Exercise 11

Name: Muhammad Sami Uddin Rafay

Roll Number: F18604013

Submitted To: Dr. Kamran Javed

Date: 02 February 2020

Practice Exercise 10

Supervised Machine Learning | KNN

Objective:

- Implement KNN using Fisher iris Data.

Equipment/Software Required:

- Python (Spyder 4.0 Anaconda Distribution)

Background:

How Does KNN Work?

1. Select no. of k of neighbors
2. Calculate the Euclidean distance of k no. of neighbors
3. Take the nearest neighbors as per the calculated Euclidean Distance
4. Among these k neighbors, count no. of data points in each category
5. Assign the new data points to that category for which the no. of neighbor is maximum
6. Our Model is ready

Code:

 # importing Libraries

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.neighbors import KNeighborsClassifier
import seaborn as sns
```

 # Loading iris dataset

```
fisher_iris=pd.read_csv(r"C:\Users\User\Documents\IrIs.csv")
```

 # Before implementing KNN on fisher_iris Data frame let's do some data quality assessment

 # Converting fisher_iris as a Data Frame

```
fisher_iris=pd.DataFrame(fisher_iris)
```

 # Printing the head of fisher_iris Data Frame

```
print("Head of Data : \n",fisher_iris.head())
```

 # Printing shape/ Dimentions fisher_iris of Data Frame

```
print("\n")
print("Shape of Data : \n",fisher_iris.shape)
```

 # Printing statistics of fisher_iris Data Frame

```
print("\n")
print("Statistics of Data : \n", fisher_iris.describe())

# Splitting the test and train Data

X = fisher_iris[["petal_length","petal_width"]]
y = fisher_iris[["species"]]

KNN=KNeighborsClassifier(n_neighbors=5)
KNN.fit(X,y)

# New Point

newpoint= [[5, 1.45]]

# New Point 2

newpoint2 = [[5, 1.45],[6,2],[2.75, 0.75]]

# Predicting New Point

Predict_NewPoint=KNN.predict(newpoint)

# Predicting New Point 2

Predict_NewPoint2=KNN.predict(newpoint2)

# Printing Predicted Results

Distances,Indexes=KNN.kneighbors(X=newpoint2, n_neighbors=5, return_distance=True)

print(Predict_NewPoint)
print(Predict_NewPoint2)
print(Distances)
print(Indexes)

# plot for first point

plt.figure(1)
iris=sns.load_dataset('iris')
sns.scatterplot(x='petal_length',y='petal_width',data=iris,hue='species')
plt.plot(5,1.5,'x', color='k')
plt.grid(True)

# plot for Second point

plt.figure(2)
iris=sns.load_dataset('iris')
sns.scatterplot(x='petal_length',y='petal_width',data=iris,hue='species')
plt.plot(5,1.45,'x', color='k')
plt.plot(6,2,'x', color='k')
plt.plot(2.75,0.75,'x', color='k')
plt.grid(True)

plt.show()
```

Output:

Head of Data:

	5.1	3.5	1.4	0.2	Iris-setosa
0	4.9	3.0	1.4	0.2	Iris-setosa
1	4.7	3.2	1.3	0.2	Iris-setosa
2	4.6	3.1	1.5	0.2	Iris-setosa
3	5.0	3.6	1.4	0.2	Iris-setosa
4	5.4	3.9	1.7	0.4	Iris-setosa

Shape of Data:

(149, 5)

Statistics of Data:

	sepal-length	sepal-width	petal-length	petal-width
count	149.000000	149.000000	149.000000	149.000000
mean	5.848322	3.051007	3.774497	1.205369
std	0.828594	0.433499	1.759651	0.761292
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.400000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

New Point 1 Prediction

['Iris-versicolor']

New Point 2 Prediction

['Iris-versicolor' 'Iris-virginica' 'Iris-versicolor']

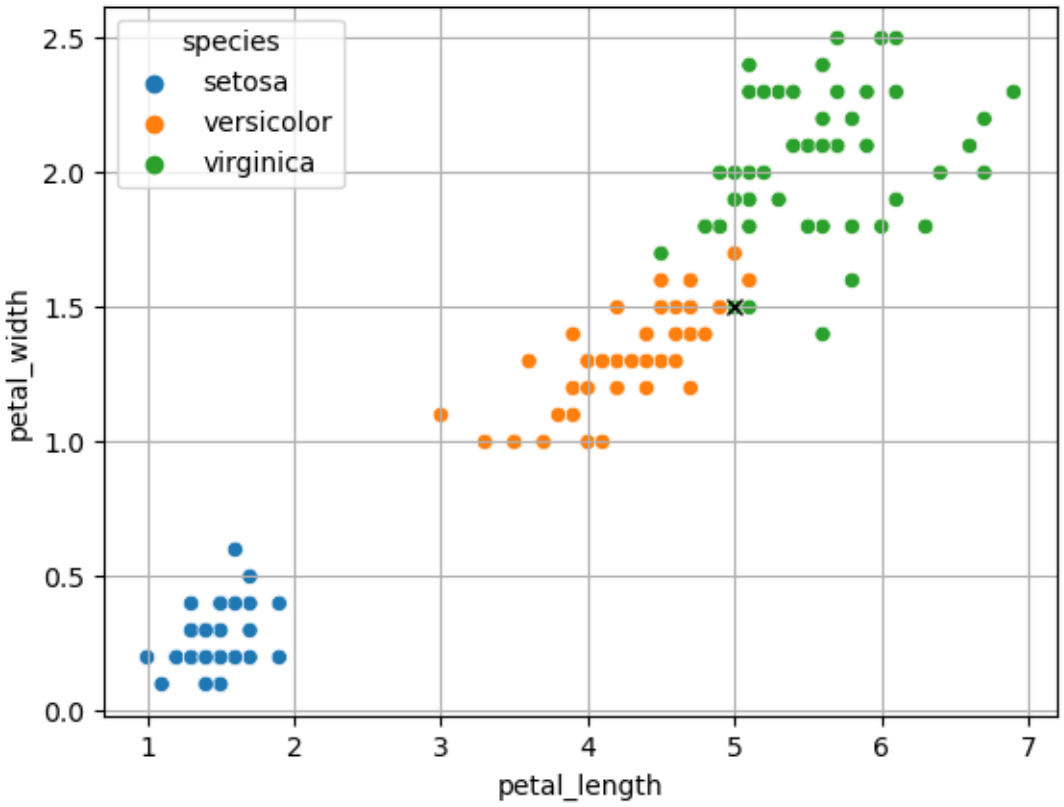
Distances

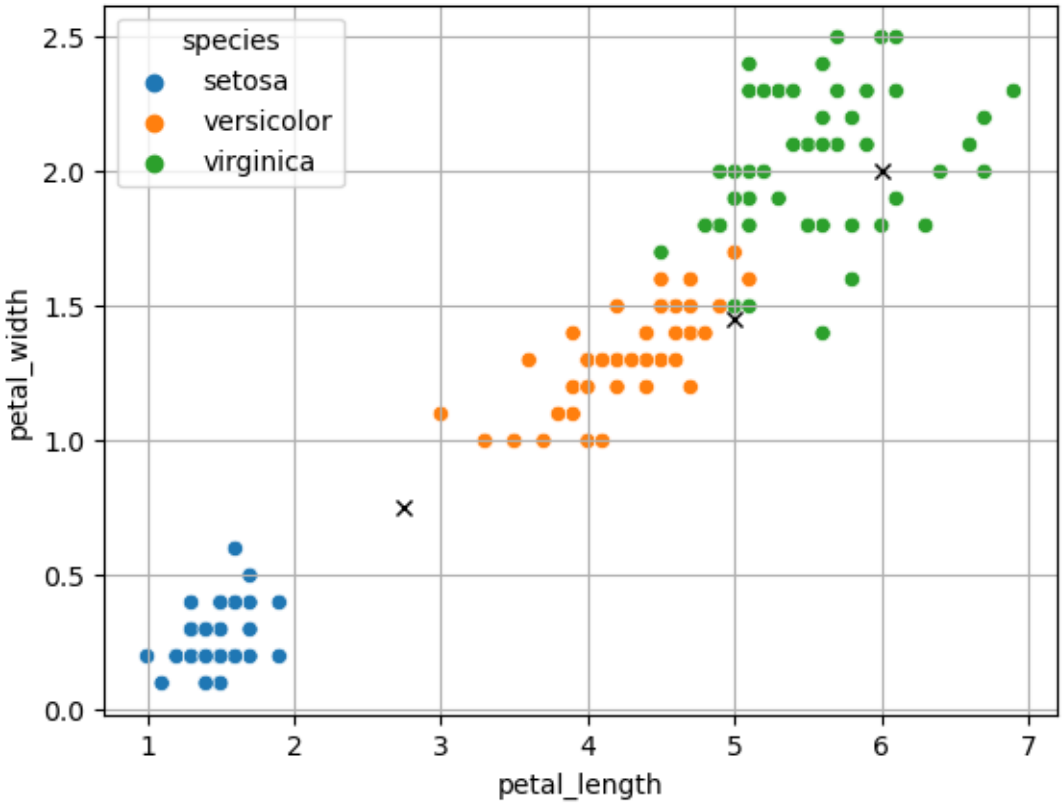
```
[[0.05  0.1118034   0.1118034   0.1118034   0.18027756]
 [0.14142136  0.14142136  0.2   0.28284271  0.28284271]
 [0.43011626  0.6041523   0.6041523   0.79056942  0.79056942]]
```

Indexes

```
[[119  52   72   133  83]
 [102  130  125  108  104]
 [ 98  57   93   79   60]]
```

Graphs:


species
setosa
versicolor
virginica
petal_width
petal_length
0.0
0.5
1.0
1.5
2.0
2.5
1
2
3
4
5
6
7


species
setosa
versicolor
virginica
petal_width
petal_length
0.0
0.5
1.0
1.5
2.0
2.5
1
2
3
4
5
6
7

Results and Discussions:

As the target attribute is “species” column of iris dataset so according to trained KNN model both of new point and new point 2 are predicted as ['Iris-versicolor' 'Iris-virginica' 'Iris-versicolor'].

Note: The starting Index of dataset in **Python** is **zero** and in **MATLAB** is **one**, so don’t confuse with indexes of predicted neighbors. Distances are predicted same in both Programming Languages.

Conclusion:

K-Nearest Neighbours is the supervised machine learning algorithm used for classification and regression. It manipulates the training data and classifies the new test data based on distance metrics. It finds the k-nearest neighbours to the test data, and then classification is performed by the majority of class labels. KNN also be used to detect anomaly by setting a threshold based on distances. We can use features of data as threshold like mean, mode, median, minimum and maximum subjected to application.