Experiment-3.1

Student Name: Ashish Kumar **UID: 23MAI10008**
Branch: ME CSE AIML **Section/Group: 23MAI-1**
Semester: 02 **Date of Performance: 27/03/2024**
Subject Name: Machine Learning Lab **Subject Code: 23CSH-651**

Aim of the Experiment :

Implementing K- Nearest Neighbor Algorithm using Python.

Theory :

K Nearest Neighbor (KNN) is a supervised learning algorithm, which is used for both regression and classification problems. KNN tries to predict the correct class for the test data by calculating the distance between the test data and all the training points. The KNN algorithm calculates the probability of the test data belonging to the classes of 'K' training data and class holds the highest probability will be selected. In the case of regression, the value is the mean of the 'K' selected training points.

KNN is a non-parametric algorithm, which means that it does not make any assumption on underlying data. KNN is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.

Distance Metrics Used in KNN Algorithm:

1) Euclidean Distance: Euclidean Distance is the cartesian distance between the two points which are in the plane/hyperplane. It can be visualized as the length of the straight line that joins the two points which are into consideration. This metric helps us calculate the net displacement done between the two states of an object.

$$d(x,y) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

2) Manhattan Distance: Manhattan Distance metric is generally used when we are interested in the total distance traveled by the object instead of the displacement. This metric is calculated by summing the absolute difference between the coordinates of the points in n-dimensions.

$$d(x, y) = |x_2 - x_1| + |y_2 - y_1|$$

Steps for implementing KNN Algorithm:

1) Select the number K which represents the number of neighbors.

2) Calculate the Euclidean distance of new data to its neighbors.

3) Take the K nearest neighbors as per the calculated Euclidean distance.

4) For classification problem, perform majority voting. The class with the most occurrences among the neighbors becomes the predicted class for the target data point.

5) For regression problem, the class label is calculated by taking average of the target values of K nearest neighbors. The calculated average value becomes the predicted output for the target data point.

Code for Experiment :

```
# Import the Libraries
import numpy as np
import pandas as pd

# Load the dataset
dataset = pd.read_csv('Iris.csv')
dataset

# Split Dataset into X and Y
X = dataset.iloc[:, [1, 2, 3, 4]].values
y = dataset.iloc[:, 5].values
```

```
# Split the X and Y Dataset into the Training set and Test set
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state = 0)

# Perform Feature Scaling as all values are not in the same range
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()

X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

# Training the K-NN model on the Training set
from sklearn.neighbors import KNeighborsClassifier
classifier = KNeighborsClassifier(n_neighbors = 5, metric = 'minkowski', p = 2)
classifier.fit(X_train, y_train)

# Predict the Test Set Results
y_pred = classifier.predict(X_test)

print("Actual values:")
print(y_test)
print("Predicted values:")
print(y_pred)

# Make the Confusion Matrix
from sklearn.metrics import confusion_matrix, accuracy_score

print("\nResults of KNN Model:")
cm = confusion_matrix(y_test, y_pred)
```

```
print("Confusion Matrix: ")
print(cm)

print("Accuracy Score: ",accuracy_score(y_test,y_pred))
```

Result/Output :

jupyter ML_Experiment8_ASHISH_23MAI10008 Last Checkpoint: a few seconds ago (autosaved)

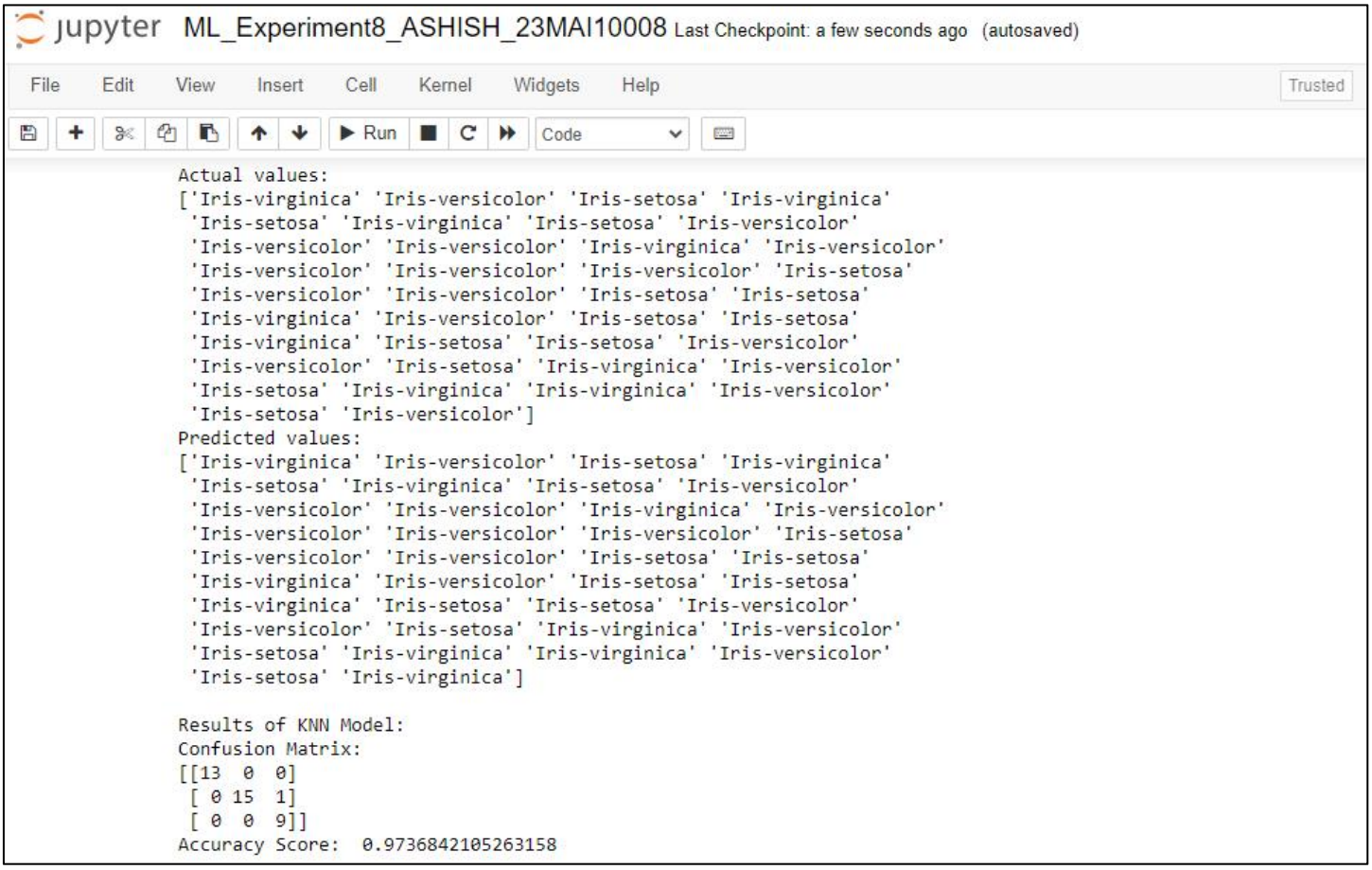
File Edit View Insert Cell Kernel Widgets Help

Run Code

Out[1]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa
...	...	...	...	...	...	...
145	146	6.7	3.0	5.2	2.3	Iris-virginica
146	147	6.3	2.5	5.0	1.9	Iris-virginica
147	148	6.5	3.0	5.2	2.0	Iris-virginica
148	149	6.2	3.4	5.4	2.3	Iris-virginica
149	150	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 6 columns



Learning outcomes (What I have learnt):

1. I learnt about various python libraries like pandas, numpy and sklearn.
2. I learnt about the concept of K nearest Neighbors Algorithm.
3. I learnt about different distance metrics used in KNN algorithm.
4. I learnt about how to select the value of K in KNN algorithm.
5. I learnt about Confusion Matrix and Accuracy Score metrics.