Fundamentals of AI and ML Lab-4

AIM : Machine Learning Classifier

1. Write a program to implement k-Nearest Neighbour algorithm to classify the iris data set. Print both correct and wrong predictions. Java/Python ML library classes can be used for this problem.
2. Write a program to implement the naïve Bayesian classifier for a sample training data set stored as a .CSV file. Compute the accuracy of the classifier, considering few test data sets.

THEORY:

K-Nearest Neighbor (KNN) is simple supervised learning algorithm used for both regression and classification problems.

Naive Bayesian Classifier is a used for classification problems and is based upon the Bayes' Theorem

In [1]:
```
# importing the libraries
import pandas as pd
from sklearn import preprocessing
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.model_selection import train_test_split
```

In [2]:
```
df=pd.read_csv("IRIS.csv")
```

In [3]:
```
df.head()
```

Out[3]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

In [4]:
```
df.shape
```

Out[4]:
```
(150, 5)
```

In [5]:
```
df.describe() #statistical measurements
```

Out[5]:

	sepal_length	sepal_width	petal_length	petal_width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667
std	0.828066	0.433594	1.764420	0.763161
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

In [6]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   sepal_length  150 non-null    float64
 1   sepal_width   150 non-null    float64
 2   petal_length  150 non-null    float64
 3   petal_width   150 non-null    float64
 4   species       150 non-null    object 
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

In [7]:
```
df.species.value_counts()
```

Out[7]:
```
Iris-virginica     50
Iris-versicolor    50
Iris-setosa        50
Name: species, dtype: int64
```

In [8]:
```
features = ['sepal_length', 'sepal_width', 'petal_length','petal_width']
x = df[features]
```

In [9]:
```
labels = df['species']
print(labels)
```

```
0         Iris-setosa
1         Iris-setosa
2         Iris-setosa
3         Iris-setosa
4         Iris-setosa
             ...
145    Iris-virginica
146    Iris-virginica
147    Iris-virginica
148    Iris-virginica
149    Iris-virginica
Name: species, Length: 150, dtype: object
```

In [10]:
```
# Splitting the data into train and test
train_x, test_x, train_y, test_y = train_test_split(x, labels, test_size=0.3)
```

In [11]:
```
# K-Nearest Neighbors Classifier
classifier = KNeighborsClassifier(n_neighbors=3)
classifier.fit(train_x,train_y)
y = classifier.predict(test_x)
```

In [12]:
```
from sklearn.metrics import accuracy_score
print("Accuracy score for KNN:", accuracy_score(test_y, y))
```

```
Accuracy score for KNN: 0.9777777777777777
```

In [13]:
```
# For KNN
print("Correct Predicition:",accuracy_score(test_y, y))
print("Worng Predicition:",(1-accuracy_score(test_y, y)))
```

```
Correct Predicition: 0.9777777777777777
Worng Predicition: 0.022222222222222254
```

In [14]:
```
# Naive Bayesian Classifier
NB = GaussianNB()
NB.fit(train_x, train_y)
y_val = NB.predict(test_x)
```

In [15]:
```
# Accuracy score for Naive Bayesian Classifier
print(accuracy_score(test_y, y_val))
```

```
0.9555555555555556
```

In [16]:
```
print("Correct Predicition:",accuracy_score(test_y, y_val))
print("Worng Predicition:",(1-accuracy_score(test_y, y_val)))
```

```
Correct Predicition: 0.9555555555555556
Worng Predicition: 0.0444444444444444
```

CONCLUSION:

K-Nearest Neighbor Classifier and Naive Bayesian Classifier were implemented in python to classify the iris dataset. The accuracy score was printed for both the classifiers. The correct and wrong predictions were also printed.