MACHINE LEARNING MINOR PROJECT

IRIS CLASSIFICATION

Problem Statement

This data set consists of the physical parameters of three species of flower — Versicolor, Setosa and Virginica. The numeric parameters which the dataset contains are Sepal width, Sepal length, Petal width and Petal length. In this data we will be predicting the classes of the flowers based on these parameters. The data consists of continuous numeric values which describe the dimensions of the respective features. We will be training the model based on these features.

LIBRARIES AND TOOLS USED

1.JUPYTER NOTEBOOK: The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text.

2.PYTHON: Python is an interpreted, object-oriented, high-level programming language. Python is one of the most versatile programming languages in the world which can easily fit any project.

3.PANDAS: Pandas offers data structure and tools for effective data manipulation and analysis. It provides facts, access to structured data. The primary instrument of Pandas is the two dimensional table consisting of column and row labels, which are called a data frame.

4.NUMPY: The NumPy library uses arrays for its inputs and outputs. It can be extended to objects for matrices.

5.MATPLOTLIB: The Matplotlib package is the most well-known library for data visualization. It is great for making graphs and plots. The graphs are also highly customizable.

6.SEABORN: It is based on Matplotlib. It's very easy to generate various plots such as heat maps, time series and box plots.

ALGORITHMS:

1.K-Nearest Neighbour: It is an algorithm for supervised learning. Where the data is 'trained' with data points corresponding to their classification. Once a point is to be predicted, it takes into account the 'K' nearest points to it to determine its classification.

K Nearest Neighbor(KNN)

In [20]:
```
from sklearn.model_selection import train_test_split
```

In [21]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=4)
print('Train set:',x_train.shape,y_train.shape)
print('Test set:',x_test.shape,y_test.shape)
```

```
Train set: (120, 4) (120,)
Test set: (30, 4) (30,)
```

In [22]:
```
from sklearn.neighbors import KNeighborsClassifier
```

In [23]:
```
k=7
neigh=KNeighborsClassifier(n_neighbors=k).fit(x_train,y_train)
print(neigh)
```

```
KNeighborsClassifier(n_neighbors=7)
```

In [24]:
```
yhat=neigh.predict(x_test)
yhat[0:5]
```

Out[24]:
```
array(['Iris-virginica', 'Iris-setosa', 'Iris-virginica',
       'Iris-virginica', 'Iris-virginica'], dtype=object)
```

In [25]:
```
from sklearn import metrics
print("Test set accuracy:",metrics.accuracy_score(y_test,yhat))
```

```
Test set accuracy: 0.9333333333333333
```

2.Decision Tree: They are build using recursive partitioning to classify data, by decreasing impurity.

Decision Tree

In [29]:
```
from sklearn.model_selection import train_test_split
```

In [30]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=4)
print('Train set:',x_train.shape,y_train.shape)
print('Test set:',x_test.shape,y_test.shape)
```

```
Train set: (120, 4) (120,)
Test set: (30, 4) (30,)
```

In [31]:
```
from sklearn.tree import DecisionTreeClassifier
```

In [32]:
```
speciesTree=DecisionTreeClassifier(criterion="entropy",max_depth=4).fit(x_train,y_train)
print(speciesTree)
```

```
DecisionTreeClassifier(criterion='entropy', max_depth=4)
```

In [33]:
```
yhat=speciesTree.predict(x_test)
yhat[0:5]
```

Out[33]:
```
array(['Iris-virginica', 'Iris-setosa', 'Iris-virginica',
       'Iris-virginica', 'Iris-virginica'], dtype=object)
```

In [34]:
```
from sklearn import metrics
print("Test set accuracy:",metrics.accuracy_score(y_test,yhat))
```

```
Test set accuracy: 0.9666666666666667
```

3.Logistic Regression: Logistic regression is a statistical and machine learning technique for classifying records of a dataset based on the values of the input fields.

Logistic Regression

In [45]:
```
from sklearn.model_selection import train_test_split
```

In [46]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=4)
print('Train set:',x_train.shape,y_train.shape)
print('Test set:',x_test.shape,y_test.shape)
```

```
Train set: (120, 4) (120,)
Test set: (30, 4) (30,)
```

In [47]:
```
from sklearn.linear_model import LogisticRegression
```

In [48]:
```
LR = LogisticRegression(multi_class='multinomial', solver='lbfgs').fit(x_train,y_train)
LR
```

Out[48]:
```
LogisticRegression(multi_class='multinomial')
```

In [49]:
```
yhat = LR.predict(x_test)
yhat
```

Out[49]:
```
array(['Iris-virginica', 'Iris-setosa', 'Iris-virginica',
       'Iris-virginica', 'Iris-virginica', 'Iris-versicolor',
       'Iris-virginica', 'Iris-setosa', 'Iris-setosa', 'Iris-versicolor',
       'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-versicolor',
       'Iris-virginica', 'Iris-setosa', 'Iris-versicolor', 'Iris-setosa',
       'Iris-setosa', 'Iris-virginica', 'Iris-setosa', 'Iris-virginica',
       'Iris-versicolor', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
       'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-virginica'],
      dtype=object)
```

```
In [50]: yhat_prob = LR.predict_proba(x_test)
         yhat_prob

Out[50]: array([[5.36003871e-05, 4.65231382e-02, 9.53423261e-01],
                [9.66889983e-01, 3.31097109e-02, 3.05680721e-07],

In [51]: from sklearn import metrics
         print("Test set accuracy:",metrics.accuracy_score(y_test,yhat))

         Test set accuracy: 0.9333333333333333
```

4.Support Vector Machine: It is a supervised algorithm that can classify cases by finding a separator. SVM works by first mapping data to a high dimensional feature space so that data points can be categorized, even when the data are not otherwise linearly separable.

Support Vector Machine

```
In [39]: from sklearn.model_selection import train_test_split

In [40]: x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=4)
         print('Train set:',x_train.shape,y_train.shape)
         print('Test set:',x_test.shape,y_test.shape)

         Train set: (120, 4) (120,)
         Test set: (30, 4) (30,)

In [42]: from sklearn import svm

In [43]: clf=svm.SVC(kernel='rbf').fit(x_train,y_train)

In [44]: yhat=clf.predict(x_test)
         yhat[0:5]

Out[44]: array(['Iris-virginica', 'Iris-setosa', 'Iris-virginica',
                'Iris-virginica', 'Iris-virginica'], dtype=object)

In [45]: from sklearn import metrics
         print("Test set accuracy:",metrics.accuracy_score(y_test,yhat))

         Test set accuracy: 0.9333333333333333
```

CONCLUSION:

Using K-Nearest Neighbour, Logistic Regression and Support Vector Machine the accuracy is **93.33%.**Using Decision Tree accuracy is **96.66%** this is a high level of accuracy and it means our model may be trustworthy enough to use. From the above models, **Decision trees** gives optimal accuracy compared to other algorithms.