Assignment on Classification using KNN.

Build an application to classify an iris flower into its specie using KNN (Iris dataset from Skleam). Display Accuracy score, classification Report & Confusion Matrix).

In [1]:

```
import pandas as pd
```

Location of dataset

In [2]:

```
url = "https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data"
```

Assign colum names to the dataset

In [3]:

```
names = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'Class']
```

Read dataset to pandas dataframe

In [4]:

```
irisdata = pd.read_csv(url, names=names)
```

In [5]:

```
irisdata.head()
```

Out[5]:

	sepal-length	sepal-width	petal-length	petal-width	Class
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 [6]:

```
irisdata.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
sepal-length    150 non-null float64
sepal-width     150 non-null float64
petal-length    150 non-null float64
petal-width     150 non-null float64
Class           150 non-null object
dtypes: float64(4), object(1)
memory usage: 5.9+ KB
```

In [7]:

```
x=irisdata.iloc[:,0:4]
```

Assign data from first fifth columns to y variable

```
In [8]: y=irisdata.select_dtypes(include=[object])
```

```
In [9]: y.head()
```

Out[9]:

	Class
0	Iris-setosa
1	Iris-setosa
2	Iris-setosa
3	Iris-setosa
4	Iris-setosa

```
In [10]: y.Class.unique()
```

```
Out[10]: array(['Iris-setosa', 'Iris-versicolor', 'Iris-virginica'], dtype=object)
```

In [11]:

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.30)
print("dimension of x-train:",x_train.shape, "x_test:",x_test.shape)
y_test
```

```
dimension of x-train: (105, 4) x_test: (45, 4)
```

Out[11]:

	Class
147	Iris-virginica
127	Iris-virginica
86	Iris-versicolor
78	Iris-versicolor
82	Iris-versicolor
60	Iris-versicolor
133	Iris-virginica
1	Iris-setosa
53	Iris-versicolor
61	Iris-versicolor
130	Iris-virginica
25	Iris-setosa
107	Iris-virginica
9	Iris-setosa
74	Iris-versicolor
104	Iris-virginica
115	Iris-virginica
108	Iris-virginica
145	Iris-virginica
68	Iris-versicolor
118	Iris-virginica
46	Iris-setosa
31	Iris-setosa
8	Iris-setosa
90	Iris-versicolor
146	Iris-virginica
114	Iris-virginica
77	Iris-versicolor
81	Iris-versicolor
11	Iris-setosa
76	Iris-versicolor
122	Iris-virginica
84	Iris-versicolor
75	Iris-versicolor
40	Iris-setosa
134	Iris-virginica

	Class
144	Iris-virginica
41	Iris-setosa
43	Iris-setosa
89	Iris-versicolor
23	Iris-setosa
54	Iris-versicolor
3	Iris-setosa
65	Iris-versicolor
48	Iris-setosa

In [15]:

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaler.fit(x_train)

x_train = scaler.transform(x_train)
x_test = scaler.transform(x_test)
```

In [16]:

```
from sklearn.neighbors import KNeighborsClassifier
```

Initialize the KNN classifier with 3 neighbors

In [17]:

```
knn = KNeighborsClassifier(n_neighbors=3)
```

Fit the model to the training data

In [18]:

```
knn.fit(x_train, y_train.values.ravel())
```

Out[18]:

```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
           metric_params=None, n_jobs=1, n_neighbors=3, p=2,
           weights='uniform')
```

In [19]:

```
predictions = knn.predict(x_test)
```

In [20]:

```
from sklearn.metrics import classification_report, confusion_matrix
print(confusion_matrix(y_test,predictions))
print(classification_report(y_test,predictions))
```

```
[[12  1  0]
 [ 0 16  1]
 [ 0  2 13]]
                 precision    recall  f1-score   support

    Iris-setosa       1.00      0.92      0.96        13
Iris-versicolor       0.84      0.94      0.89        17
 Iris-virginica       0.93      0.87      0.90        15

    avg / total       0.92      0.91      0.91        45
```

In [ ]:

```
```