Implementation of Random Forest Algorithm

Group-10

Aryan Pandey (1906465)

Hari Om (1906476)

Samim Hossain Mondal (1906500)

Souhardya Pal (1906511)

Tapabrata Roy (1906522)

In [1]:

```
#importing required libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [2]:

```
#reading the given data set
iris = pd.read_csv("C:\\Users\\KIIT\\Documents\\3rd Year\\6th SEM\\Machine Learning\\M2\\ir
iris
```

Out[2]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	virginica
146	6.3	2.5	5.0	1.9	virginica
147	6.5	3.0	5.2	2.0	virginica
148	6.2	3.4	5.4	2.3	virginica
149	5.9	3.0	5.1	1.8	virginica

150 rows × 5 columns

In [3]:

```
#checking the datatypes of the dataset
iris.dtypes
```

Out[3]:

```
sepal_length    float64
sepal_width     float64
petal_length    float64
petal_width     float64
species          object
dtype: object
```

In [4]:

```
#splitting the dataset into two parts; one part contains the float variables and another on
x = iris.iloc[:,:-1].values
y = iris.iloc[:, -1].values
```

In [5]:

```
print(x)
```

```
[[5.1 3.5 1.4 0.2]
 [4.9 3.  1.4 0.2]
 [4.7 3.2 1.3 0.2]
 [4.6 3.1 1.5 0.2]
 [5.  3.6 1.4 0.2]
 [5.4 3.9 1.7 0.4]
 [4.6 3.4 1.4 0.3]
 [5.  3.4 1.5 0.2]
 [4.4 2.9 1.4 0.2]
 [4.9 3.1 1.5 0.1]
 [5.4 3.7 1.5 0.2]
 [4.8 3.4 1.6 0.2]
 [4.8 3.  1.4 0.1]
 [4.3 3.  1.1 0.1]
 [5.8 4.  1.2 0.2]
 [5.7 4.4 1.5 0.4]
 [5.4 3.9 1.3 0.4]
 [5.1 3.5 1.4 0.3]
 [5.7 3.8 1.7 0.3]
```

In [6]:

```
print(y)
```

```
['setosa' 'setosa' 'setosa' 'setosa' 'setosa' 'setosa' 'setosa' 'setosa'
 'setosa' 'setosa' 'setosa' 'setosa' 'setosa' 'setosa' 'setosa' 'setosa'
 'setosa' 'setosa' 'setosa' 'setosa' 'setosa' 'setosa' 'setosa' 'setosa'
 'setosa' 'setosa' 'setosa' 'setosa' 'setosa' 'setosa' 'setosa' 'setosa'
 'setosa' 'setosa' 'setosa' 'setosa' 'setosa' 'setosa' 'setosa' 'setosa'
 'setosa' 'setosa' 'setosa' 'setosa' 'setosa' 'setosa' 'setosa' 'setosa'
 'setosa' 'setosa' 'versicolor' 'versicolor' 'versicolor' 'versicolor'
 'versicolor' 'versicolor' 'versicolor' 'versicolor' 'versicolor'
 'versicolor' 'versicolor' 'versicolor' 'versicolor' 'versicolor'
 'versicolor' 'versicolor' 'versicolor' 'versicolor' 'versicolor'
 'versicolor' 'versicolor' 'versicolor' 'versicolor' 'versicolor'
 'versicolor' 'versicolor' 'versicolor' 'versicolor' 'versicolor'
 'versicolor' 'versicolor' 'versicolor' 'versicolor' 'versicolor'
 'versicolor' 'versicolor' 'versicolor' 'versicolor' 'versicolor'
 'versicolor' 'versicolor' 'versicolor' 'versicolor' 'versicolor'
 'versicolor' 'versicolor' 'versicolor' 'versicolor' 'versicolor'
 'versicolor' 'virginica' 'virginica' 'virginica' 'virginica' 'virginica'
 'virginica' 'virginica' 'virginica' 'virginica' 'virginica' 'virginica'
 'virginica' 'virginica' 'virginica' 'virginica' 'virginica' 'virginica'
 'virginica' 'virginica' 'virginica' 'virginica' 'virginica' 'virginica'
 'virginica' 'virginica' 'virginica' 'virginica' 'virginica' 'virginica'
 'virginica' 'virginica' 'virginica' 'virginica' 'virginica' 'virginica'
 'virginica' 'virginica' 'virginica' 'virginica' 'virginica' 'virginica'
 'virginica' 'virginica' 'virginica' 'virginica' 'virginica' 'virginica'
 'virginica' 'virginica' 'virginica']
```

In [7]:

```
print("Length of x =",len(x))
print("Length of y =",len(y))
```

```
Length of x = 150
Length of y = 150
```

In [8]:

```
#Encoding the categorical dependent variable
from sklearn.preprocessing import LabelEncoder
label = LabelEncoder()
y = label.fit_transform(y)
```

In [9]:

```
print(y)
```

```
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2]
```

In [10]:

```
#splitting into train and test dataset
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_state = 1
```

In [11]:

```
print(x_train)
```

```
 [7.3 2.9 6.3 1.8]
 [5.8 2.7 5.1 1.9]
 [6.3 2.5 5.  1.9]
 [4.4 3.2 1.3 0.2]
 [5.6 3.  4.5 1.5]
 [5.  3.4 1.5 0.2]
 [5.  3.6 1.4 0.2]
 [5.7 2.8 4.1 1.3]
 [5.2 3.4 1.4 0.2]
 [7.9 3.8 6.4 2. ]
 [6.1 2.6 5.6 1.4]
 [5.6 2.8 4.9 2. ]
 [6.1 2.9 4.7 1.4]
 [4.6 3.6 1.  0.2]
 [5.  3.5 1.3 0.3]
 [7.2 3.2 6.  1.8]
 [4.5 2.3 1.3 0.3]
 [5.9 3.2 4.8 1.8]
 [5.5 2.3 4.  1.3]
 [6.5 3.  5.8 2.2]
```

In [12]:

```
print(x_test)
```

```
[[4.6 3.4 1.4 0.3]
 [5.9 3.  4.2 1.5]
 [5.5 2.6 4.4 1.2]
 [4.8 3.1 1.6 0.2]
 [5.4 3.4 1.5 0.4]
 [5.7 2.8 4.5 1.3]
 [7.2 3.6 6.1 2.5]
 [5.1 2.5 3.  1.1]
 [5.2 2.7 3.9 1.4]
 [6.3 2.9 5.6 1.8]
 [6.2 2.8 4.8 1.8]
 [6.2 2.9 4.3 1.3]
 [6.4 2.9 4.3 1.3]
 [5.8 2.7 4.1 1. ]
 [6.4 3.2 5.3 2.3]
 [5.  3.  1.6 0.2]
 [5.7 3.  4.2 1.2]
 [6.3 3.3 6.  2.5]
 [4.9 3.1 1.5 0.1]
 [7.2 3.  5.8 1.6]
 [6.1 2.8 4.  1.3]
 [4.6 3.2 1.4 0.2]
 [6.7 3.1 4.7 1.5]
 [4.9 2.4 3.3 1. ]
 [5.3 3.7 1.5 0.2]
 [5.4 3.9 1.7 0.4]
 [6.8 3.  5.5 2.1]
 [6.2 3.4 5.4 2.3]
 [7.7 2.8 6.7 2. ]
 [6.  2.9 4.5 1.5]]
```

In [13]:

```
print(y_train)
```

```
[0 2 1 2 0 0 2 0 0 1 2 0 0 1 2 2 0 1 0 2 2 2 2 1 1 1 0 1 2 2 2 2 0 2 1 2 1
 0 1 0 0 2 1 1 1 1 0 2 2 2 0 1 0 0 2 2 1 1 2 1 0 0 1 1 1 0 2 0 1 0 1 0 2 2
 2 0 1 0 0 1 0 2 2 2 1 0 0 2 0 1 1 2 1 1 1 0 0 2 1 0 2 1 0 2 1 0 0 0 0 2 0
 1 0 0 2 1 2 2 2 2]
```

In [14]:

```
print(y_test)
```

```
[0 1 1 0 0 1 2 1 1 2 2 1 1 1 2 0 1 2 0 2 1 0 1 1 0 0 2 2 2 1]
```

In [15]:

```
print("Length of x_train =",len(x_train))
print("Length of x_test =",len(x_test))
print("Length of y_train =",len(y_train))
print("Length of y_test =",len(y_test))
```

```
Length of x_train = 120
Length of x_test = 30
Length of y_train = 120
Length of y_test = 30
```

In [16]:

```
#defining the random forest classifier
from sklearn.ensemble import RandomForestClassifier
#create classifier object
clf = RandomForestClassifier(n_estimators = 5, criterion = 'gini', random_state = 15)
```

In [17]:

```
#fit the classifier and training the random forest classifier
clf.fit(x_train, y_train)
```

Out[17]:

```
RandomForestClassifier(n_estimators=5, random_state=15)
```

In [18]:

```
#predicting the test result
predict = clf.predict(x_test)
print(predict)
```

```
[0 1 1 0 0 1 2 1 1 2 2 1 1 1 2 0 1 2 0 1 1 0 1 1 0 0 2 2 2 1]
```

In [19]:

```
predict_vertical = predict.reshape(len(predict),1)
print(predict_vertical)
```

```
[[0]
 [1]
 [1]
 [0]
 [0]
 [1]
 [2]
 [1]
 [1]
 [2]
 [2]
 [1]
 [1]
 [1]
 [2]
 [0]
 [1]
 [2]
 [0]
 [1]
 [1]
 [0]
 [1]
 [1]
 [0]
 [0]
 [2]
 [2]
 [2]
 [1]]
```

In [20]:

```
original_vertical = y_test.reshape(len(y_test),1)
print(original_vertical)
```

```
[[0]
 [1]
 [1]
 [0]
 [0]
 [1]
 [2]
 [1]
 [1]
 [2]
 [2]
 [1]
 [1]
 [1]
 [2]
 [0]
 [1]
 [2]
 [0]
 [2]
 [1]
 [0]
 [1]
 [1]
 [0]
 [0]
 [2]
 [2]
 [2]
 [1]]
```

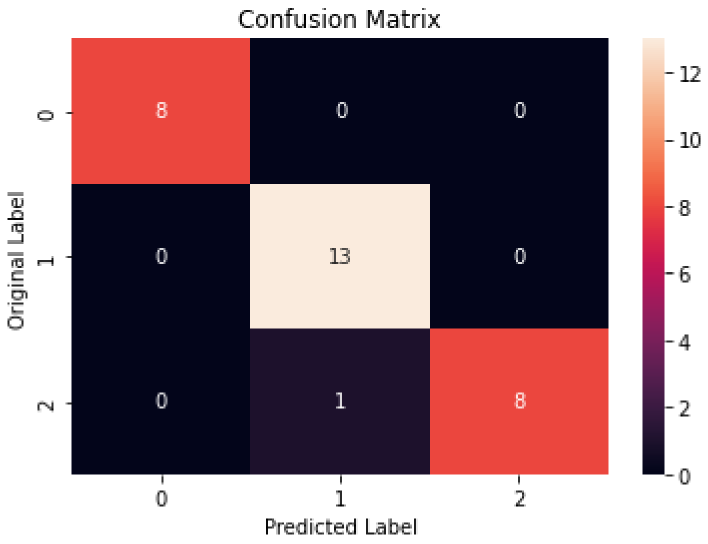
In [21]:

```
original_predict = np.concatenate((original_vertical, predict_vertical), axis = 1)
print(original_predict)
```

```
[[0 0]
 [1 1]
 [1 1]
 [0 0]
 [0 0]
 [1 1]
 [2 2]
 [1 1]
 [1 1]
 [2 2]
 [2 2]
 [1 1]
 [1 1]
 [1 1]
 [2 2]
 [0 0]
 [1 1]
 [2 2]
 [0 0]
 [2 1]
 [1 1]
 [0 0]
 [1 1]
 [1 1]
 [0 0]
 [0 0]
 [2 2]
 [2 2]
 [2 2]
 [1 1]]
```

In [22]:

```
#plotting confusion matrix
from sklearn.metrics import confusion_matrix
sns.heatmap(confusion_matrix(y_test, predict), annot = True)
plt.xlabel("Predicted Label")
plt.ylabel("Original Label")
plt.title("Confusion Matrix")
plt.show()
```



In [23]:

```
#Finding the classifier Accuracy
from sklearn.metrics import accuracy_score
accuracy_score(y_test, predict)
```

Out[23]:

```
0.9666666666666667
```

In [24]:

```
#evaluating the classifier on new data
prediction = clf.predict([[5, 3, 1.6, 0.2]])
print(prediction)
```

```
[0]
```