KNN

In [7]:
```
import numpy as np
import pandas as pd
```

In [8]:
```
df = pd.read_csv('C:/Users/SW20407278/Desktop/Final AI/Hands-On/Classification/iris.cs
```

In [9]:
```
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   variety       150 non-null    object 
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

In [10]:
```
### To check the dataset is balanced or not
df.variety.value_counts()
```

Out[10]:
```
Setosa        50
Versicolor    50
Virginica     50
Name: variety, dtype: int64
```

In [11]:
```
df.head()
```

Out[11]:

	sepal.length	sepal.width	petal.length	petal.width	variety
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

In [12]:
```
## Features and Label
features = df.iloc[:,:-1].values
label = df.iloc[:,4].values
```

In [13]:
```
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
```

In [14]:
```
## Creation of Train Test Split
xtrain,xtest,ytrain,ytest = train_test_split(features,label,test_size=0.2,random_state
```

In [15]:
```
## Model building KNN
```

```
model_KNN = KNeighborsClassifier(n_neighbors=5)
model_KNN.fit(xtrain,ytrain)
```

Out[15]:
```
KNeighborsClassifier()
```

In [16]:
```
## Training and Testing Accuracy
print(model_KNN.score(xtrain,ytrain))
print(model_KNN.score(xtest,ytest))
```

```
0.9583333333333334
1.0
```

In [17]:
```
## Confusion Matrix
from sklearn.metrics import confusion_matrix
confusion_matrix(label,model_KNN.predict(features))
```

Out[17]:
```
array([[50,  0,  0],
       [ 0, 47,  3],
       [ 0,  2, 48]], dtype=int64)
```

In [18]:
```
from sklearn.metrics import classification_report
print(classification_report(label,model_KNN.predict(features)))
```

```
              precision    recall  f1-score   support

      Setosa       1.00      1.00      1.00        50
  Versicolor       0.96      0.94      0.95        50
   Virginica       0.94      0.96      0.95        50

    accuracy                           0.97       150
   macro avg       0.97      0.97      0.97       150
weighted avg       0.97      0.97      0.97       150
```

In [ ]:
```

```

In [ ]:
```

```