In [1]:
```python
import numpy as np
import pandas as pd
```

In [2]:
```python
df=pd.read_csv('Iris.csv')
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 [3]:
```python
df.variety.value_counts()
```

Out[3]:
```
variety
Setosa        50
Versicolor    50
Virginica     50
Name: count, dtype: int64
```

In [4]:
```python
df.head()
```

Out[4]:

	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 [5]:
```python
features=df.iloc[:,:-1].values
label=df.iloc[:,4].values
```

In [6]:
```python
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
```

In [8]:
```python
xtrain,xtest,ytrain,ytest=train_test_split(features,label,test_size=.2,random_state
model_KNN=KNeighborsClassifier(n_neighbors=5)
model_KNN.fit(xtrain,ytrain)
```

Out[8]:

```
▾ KNeighborsClassifier

▶ Parameters
```

In [9]:

```python
print(model_KNN.score(xtrain,ytrain))
print(model_KNN.score(xtest,ytest))
```

```
0.9666666666666667
1.0
```

In [10]:

```python
from sklearn.metrics import confusion_matrix
confusion_matrix(label,model_KNN.predict(features))
```

Out[10]:

```
array([[50,  0,  0],
       [ 0, 47,  3],
       [ 0,  1, 49]])
```

In [11]:

```python
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       0.98      0.94      0.96        50
   Virginica       0.94      0.98      0.96        50

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

In [ ]:

```python

```