Classifying Iris Species with k-nearest neighbours.

Dataset is imported from scikit learn.

In [46]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import scipy as sp
import IPython, sklearn,sys
import seaborn as sns
```

Importing and loading dataset

In [10]:
```
from sklearn.datasets import load_iris
iris_dataset=load_iris()
```

In [31]:
```
print("Keys of iris_dataset: \n{}".format(iris_dataset.keys()))
print(iris_dataset['DESCR'][:200])
print(iris_dataset['data'][:20])
print(iris_dataset['target'])
print(iris_dataset['target_names'])
print(iris_dataset['feature_names'])
```

```
Keys of iris_dataset:
dict_keys(['data', 'target', 'target_names', 'DESCR', 'feature_names'])
Iris Plants Database
====================

Notes
-----
Data Set Characteristics:
    :Number of Instances: 150 (50 in each of three classes)
    :Number of Attributes: 4 numeric, predictive attributes
[[ 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]
 [ 5.1  3.8  1.5  0.3]]
[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 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 2]
['setosa' 'versicolor' 'virginica']
['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width
(cm)']
```

There are 150 rows, of which each row represents a flower (called the sample) and the columns represent the four measurements (called the features):

In [29]:
```
iris_dataset['data'].shape
```

Out[29]: (150, 4)

#The train_test_split function shuffles and splits the dataset- 75% for training & 25% for testing

the output is four NumPy arrays, two data and two target arrays- one for taining and one for testing

In [37]:
```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(iris_dataset['data'],iris_datas
et['target'],random_state=0)
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(112, 4)
(38, 4)
(112,)
(38,)
```

import and connfigure k-nearest neighbor's estimator class

In [72]:
```
from sklearn.neighbors import KNeighborsClassifier
knn= KNeighborsClassifier(n_neighbors=1)
```

In [73]:
```
knn.fit(X_train, y_train)
```

Out[73]:
```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
           metric_params=None, n_jobs=1, n_neighbors=1, p=2,
           weights='uniform')
```

Making Predictions

In [74]:
```
New_flower = np.array([[5, 2.9, 1, 0.2]])
prediction = knn.predict(New_flower)
print("its a type {} flower".format(prediction))
print(iris_dataset['target_names'][prediction])
```

```
its a type [0] flower
['setosa']
```

Validating the model

In [75]:
```
y_pred = knn.predict(X_test)

print(y_pred)

print(iris_dataset['target_names'][y_pred])
```

```
[2 1 0 2 0 2 0 1 1 1 2 1 1 1 1 0 1 1 0 0 2 1 0 0 2 0 0 1 1 0 2 1 0 2 2 1 0
 2]
['virginica' 'versicolor' 'setosa' 'virginica' 'setosa' 'virginica'
 'setosa' 'versicolor' 'versicolor' 'versicolor' 'virginica' 'versicolor'
 'versicolor' 'versicolor' 'versicolor' 'setosa' 'versicolor' 'versicolor'
 'setosa' 'setosa' 'virginica' 'versicolor' 'setosa' 'setosa' 'virginica'
 'setosa' 'setosa' 'versicolor' 'versicolor' 'setosa' 'virginica'
 'versicolor' 'setosa' 'virginica' 'virginica' 'versicolor' 'setosa'
 'virginica']
```

Calculating the accuracy

In [83]:
```
print(knn.score(X_test, y_test))

print("The model scored {:.2f}% on the test dataset." .format(knn.score(X_test
, y_test)))
```

```
0.973684210526
The model scored 0.97% on the test dataset.
```