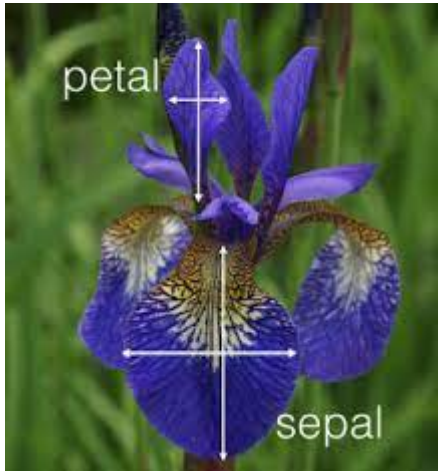
K-means on IRIS Dataset



- Importing important Libraries

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.patches as mpatches
from sklearn.datasets import load_iris
from sklearn.cluster import KMeans
import sklearn.metrics as sm

import warnings
warnings.filterwarnings("ignore")
%matplotlib inline
```

- Loading data

In [2]:
```
iris_data = load_iris()
```

In [3]:
```
print(iris_data.data)
```

```
[[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]
```

- Shape of given data

In [4]:
```
print(iris_data.data.shape)
```

```
(150, 4)
```

In [5]:
```
print(iris_data.target_names)
```

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

In [6]:
```
print(iris_data.target)
```

```
[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]
```

Data Preparation

In [7]:
```
# x = pd.DataFrame(iris_data.data, columns=['Sepal Length', 'Sepal Width', 'Petal Length', 'Petal Width'])
x = pd.DataFrame(iris_data.data , columns = iris_data.feature_names)
```

In [8]:
```
1 x.head()
```

Out[8]:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2

In [9]:
```
1 print(x.describe())
```

```
       sepal length (cm)  sepal width (cm)  petal length (cm)  \
count         150.000000        150.000000         150.000000
mean            5.843333          3.057333           3.758000
std             0.828066          0.435866           1.765298
min             4.300000          2.000000           1.000000
25%             5.100000          2.800000           1.600000
50%             5.800000          3.000000           4.350000
75%             6.400000          3.300000           5.100000
max             7.900000          4.400000           6.900000

       petal width (cm)
count        150.000000
mean           1.199333
std            0.762238
min            0.100000
25%            0.300000
50%            1.300000
75%            1.800000
max            2.500000
```

In [10]:
```
1 print(x.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 4 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   sepal length (cm)  150 non-null    float64
 1   sepal width (cm)   150 non-null    float64
 2   petal length (cm)  150 non-null    float64
 3   petal width (cm)   150 non-null    float64
dtypes: float64(4)
memory usage: 4.8 KB
None
```

In [11]:
```
1 print(x.isnull().sum())
```

```
sepal length (cm)    0
sepal width (cm)     0
petal length (cm)    0
petal width (cm)     0
dtype: int64
```

In [12]:
```
1 x.isnull().values.any()
```

Out[12]:
```
False
```

In [13]:
```
1 y = pd.DataFrame(iris_data.target, columns=['Species'])
```

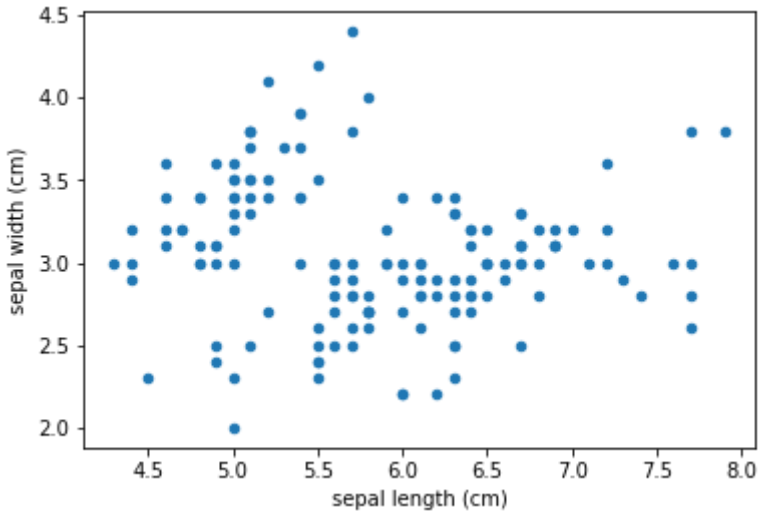
In [14]:
```
1 y.head()
```

Out[14]:

	Species
0	0
1	0
2	0
3	0
4	0

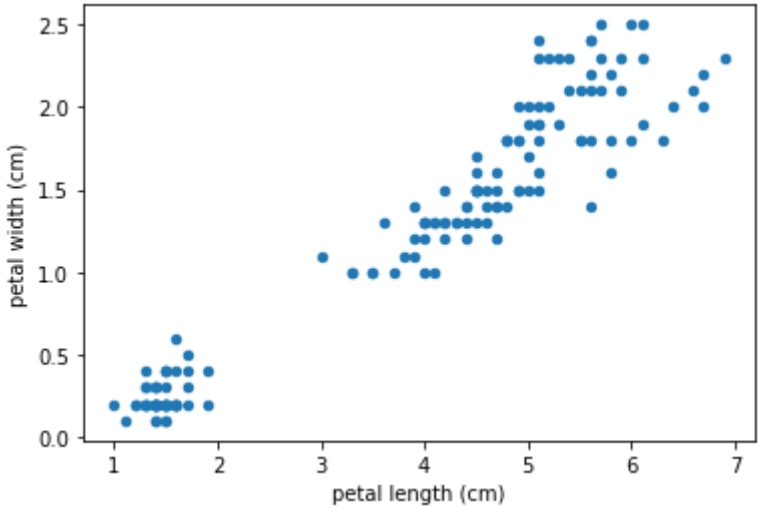
EDA

In [15]:
```
1 x.plot(kind="scatter", x="sepal length (cm)",   y="sepal width (cm)")
2 plt.show()
```



In [16]:

```
x.plot(kind="scatter", x="petal length (cm)",   y="petal width (cm)")
plt.show()
```

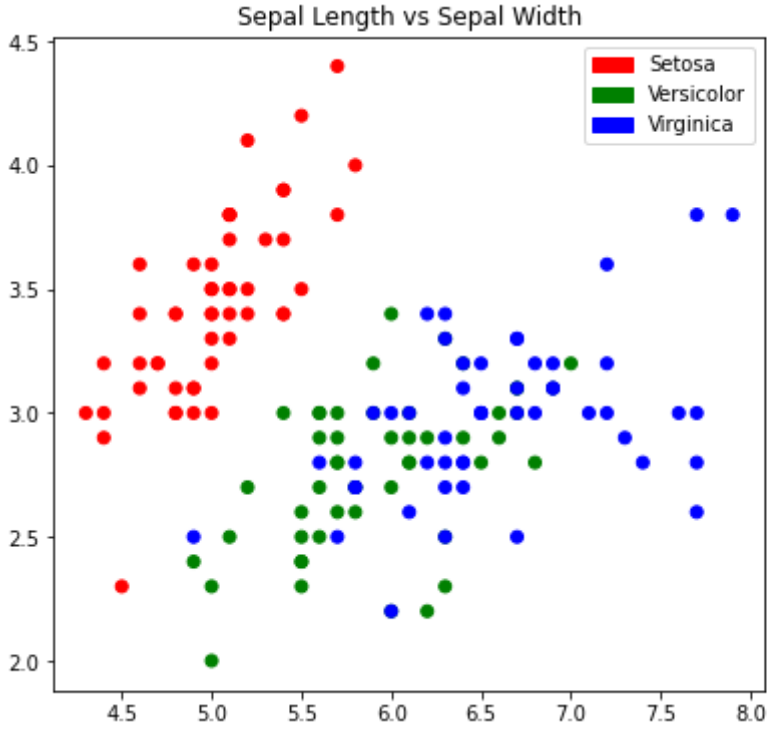


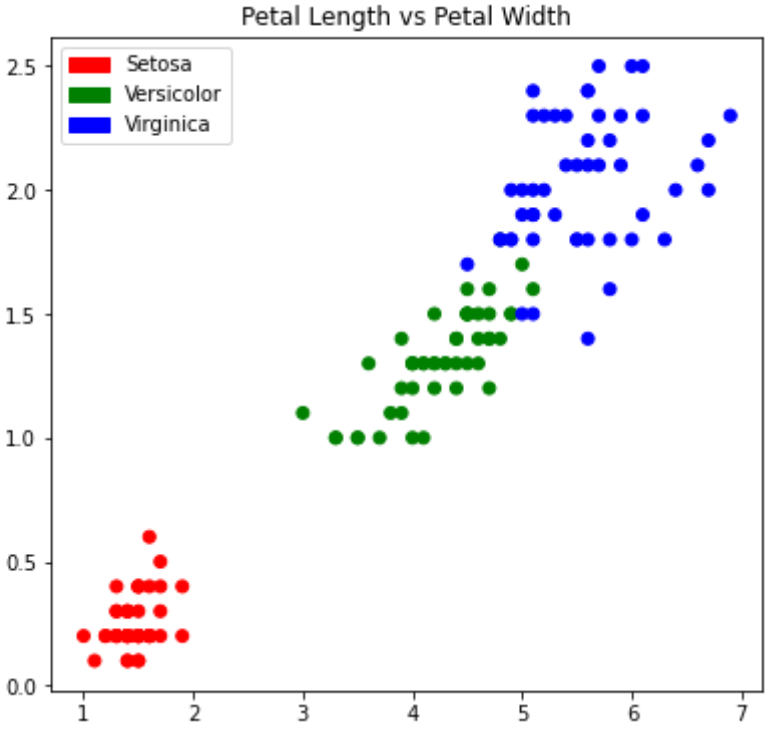
In [17]:

```
plt.figure(figsize=(14,6))
colors = np.array(['red', 'green', 'blue'])
targets_legend = np.array(iris_data.target_names)

# ['setosa' 'versicolor' 'virginica']
red_patch = mpatches.Patch(color='red', label='Setosa')
green_patch = mpatches.Patch(color='green', label='Versicolor')
blue_patch = mpatches.Patch(color='blue', label='Virginica')


plt.subplot(1, 2, 1)
plt.scatter(x['sepal length (cm)'], x['sepal width (cm)'], c=colors[y['Species']])
plt.title('Sepal Length vs Sepal Width')
plt.legend(handles=[red_patch, green_patch, blue_patch])

plt.subplot(1,2,2)
plt.scatter(x['petal length (cm)'], x['petal width (cm)'], c= colors[y['Species']])
plt.title('Petal Length vs Petal Width')
plt.legend(handles=[red_patch, green_patch, blue_patch])
```

Out[17]:

```
<matplotlib.legend.Legend at 0x1a779d79f70>
```





Model Training

In [18]:
```
kmeans_model = KMeans(n_clusters=3, random_state=1)
y_kmeans = kmeans_model.fit_predict(x)
print(y_kmeans)
```

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

Predicted Centroids

In [19]:
```
1 print(kmeans_model.cluster_centers_)
```

```
[[5.9016129  2.7483871  4.39354839 1.43387097]
 [5.006      3.428      1.462      0.246     ]
 [6.85       3.07368421 5.74210526 2.07105263]]
```

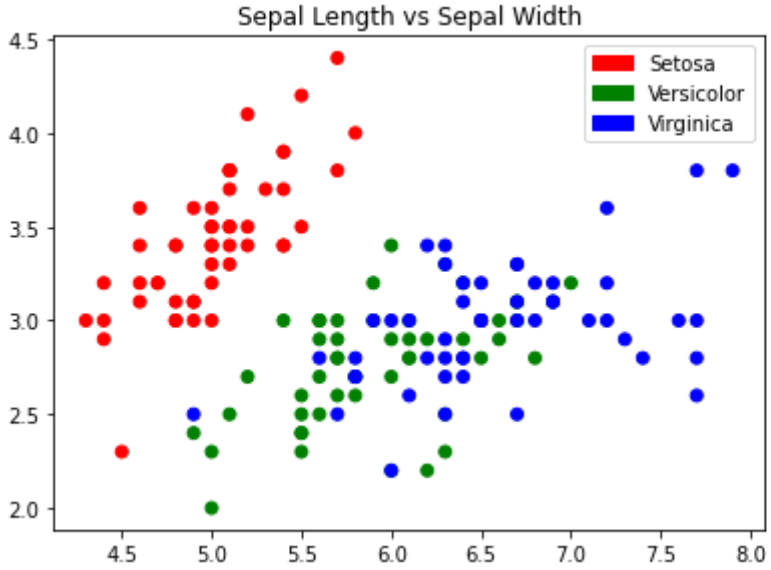
Model Evaluation

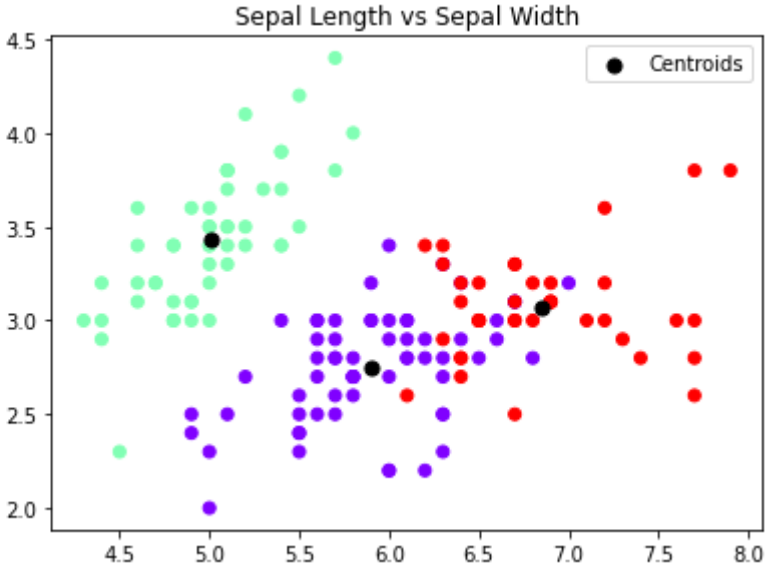
In [20]:

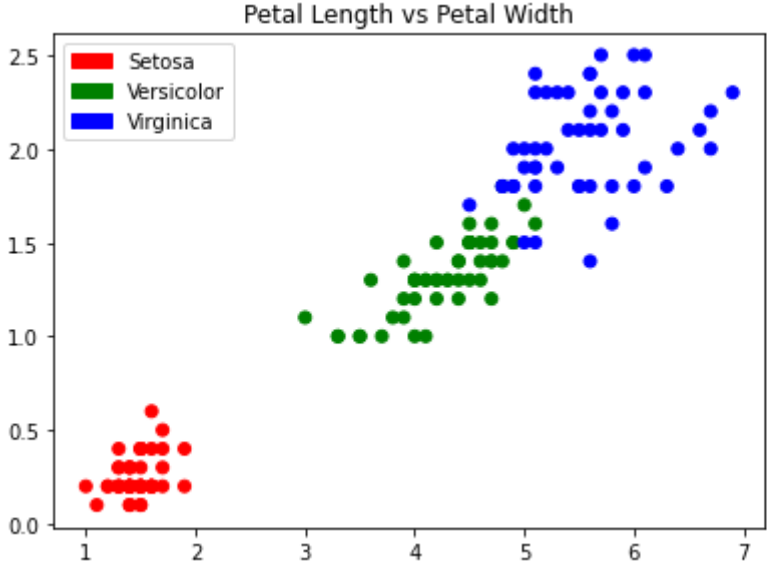
```
plt.figure(figsize=(14,10))

colors = np.array(['red', 'green', 'blue'])
targets_legend = np.array(iris_data.target_names)

# ['setosa' 'versicolor' 'virginica']
red_patch = mpatches.Patch(color='red', label='Setosa')
green_patch = mpatches.Patch(color='green', label='Versicolor')
blue_patch = mpatches.Patch(color='blue', label='Virginica')


plt.subplot(2, 2, 1)
plt.scatter(x['sepal length (cm)'], x['sepal width (cm)'], c=colors[y['Species']])
plt.title('Sepal Length vs Sepal Width')
plt.legend(handles=[red_patch, green_patch, blue_patch])

# Predicted Cluster for sepal length and width
plt.subplot(2, 2, 2)
plt.scatter(x['sepal length (cm)'], x['sepal width (cm)'], c= y_kmeans, cmap='rainbow')
plt.scatter(kmeans_model.cluster_centers_[:,0], kmeans_model.cluster_centers_[:,1],s = 50, c = 'black', label = '
plt.title('Sepal Length vs Sepal Width')
plt.legend()

plt.subplot(2, 2, 3)
plt.scatter(x['petal length (cm)'], x['petal width (cm)'], c= colors[y['Species']])
plt.title('Petal Length vs Petal Width')
plt.legend(handles=[red_patch, green_patch, blue_patch])

# Predicted Cluster for petal length and width
plt.subplot(2, 2, 4)
plt.scatter(x['petal length (cm)'], x['petal width (cm)'], c= y_kmeans, cmap='rainbow')
plt.scatter(kmeans_model.cluster_centers_[:,2], kmeans_model.cluster_centers_[:,3],s = 50, c = 'black', label = '
plt.title('Petal Length vs Petal Width')
plt.legend()
```

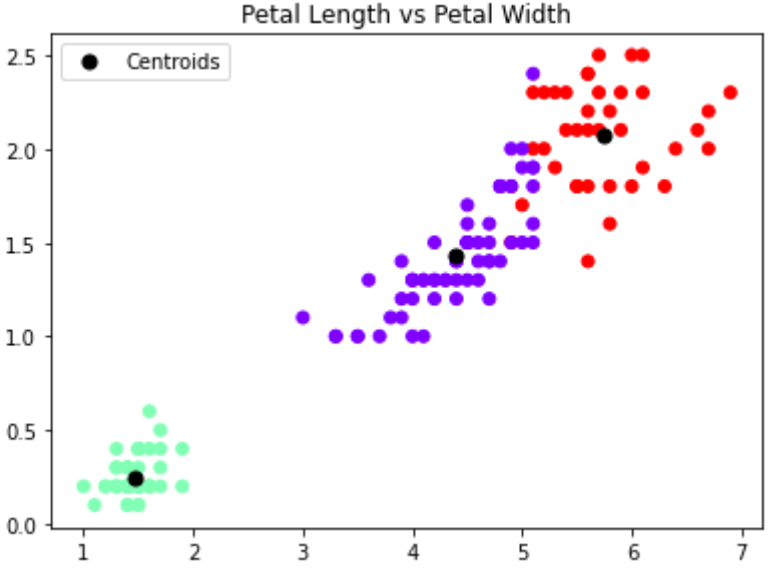
Out[20]:

```
<matplotlib.legend.Legend at 0x1a77a5dc760>
```


Sepal Length vs Sepal Width
Setosa
Versicolor
Virginica
4.5
4.0
3.5
3.0
2.5
2.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0

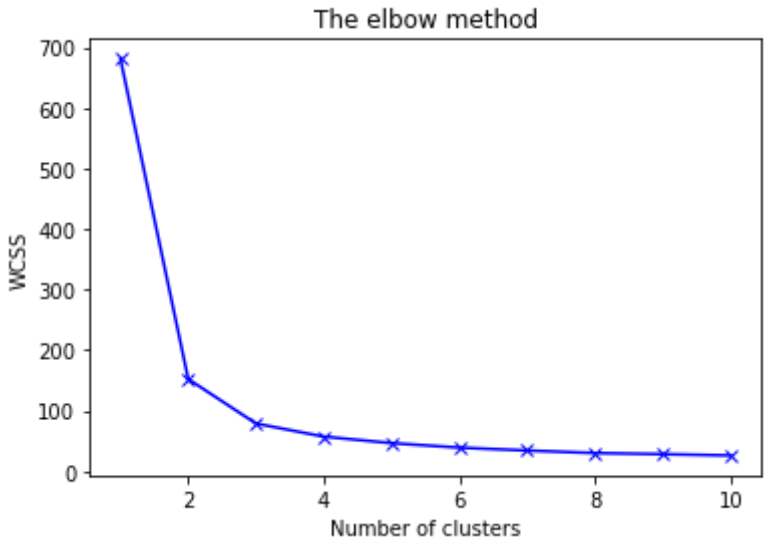

Sepal Length vs Sepal Width
Centroids
4.5
4.0
3.5
3.0
2.5
2.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0


Petal Length vs Petal Width
Setosa
Versicolor
Virginica
2.5
2.0
1.5
1.0
0.5
0.0
1
2
3
4
5
6
7


Petal Length vs Petal Width
Centroids
2.5
2.0
1.5
1.0
0.5
0.0
1
2
3
4
5
6
7

In [21]:

```
from sklearn.cluster import KMeans
wcss = []

for i in range(1, 11):
    kmeans = KMeans(n_clusters = i, random_state = 0)
    kmeans.fit(x)
    wcss.append(kmeans.inertia_)

#Plotting the results onto a line graph, allowing us to observe 'The elbow'
plt.plot(range(1, 11), wcss ,'bx-')
plt.title('The elbow method')
plt.xlabel('Number of clusters')
#within cluster sum of squares
plt.ylabel('WCSS')
plt.show()
```



In [ ]:

```
1
```