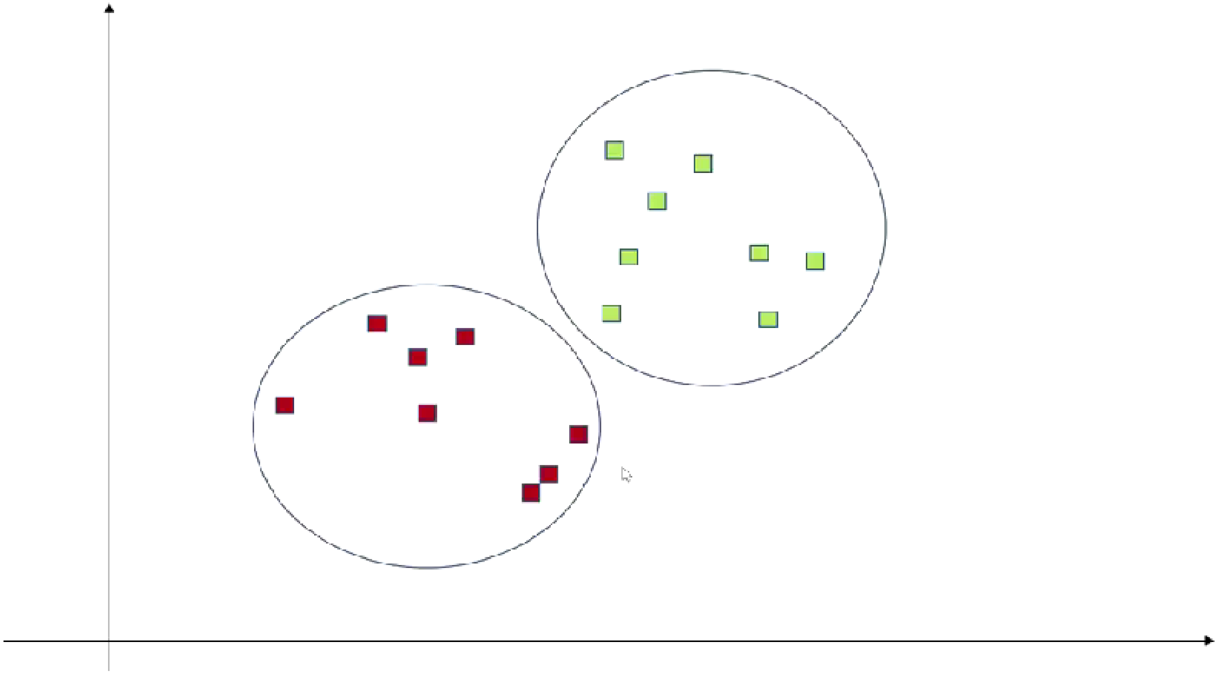
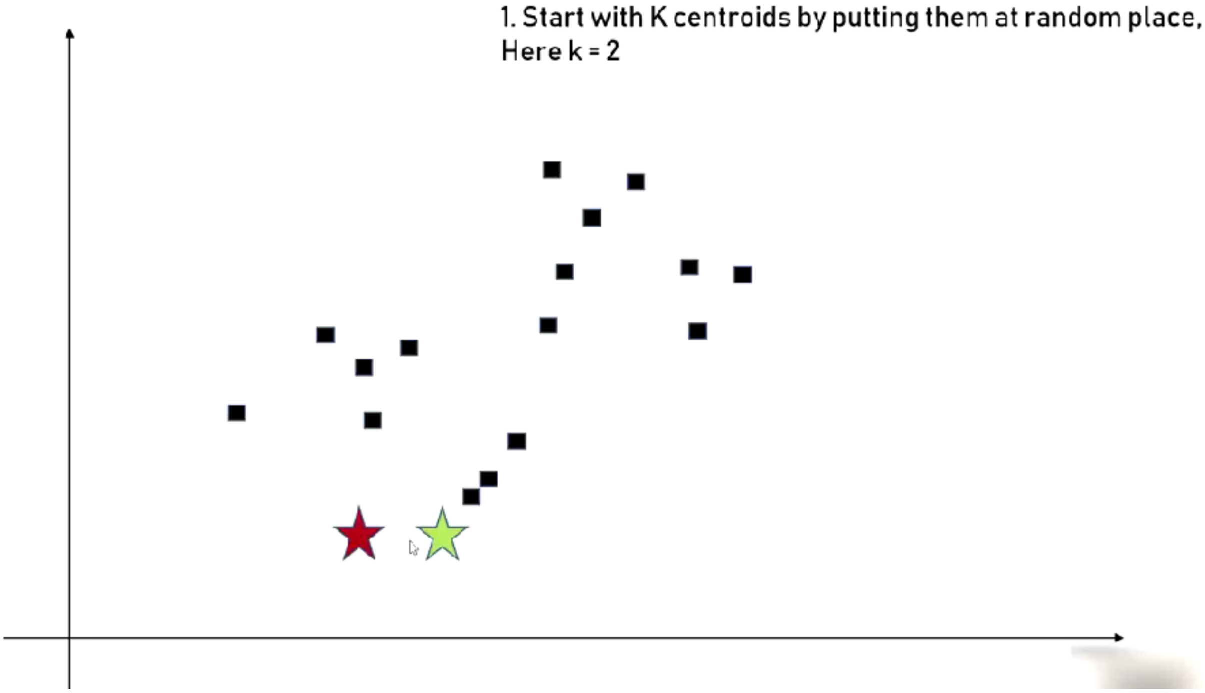
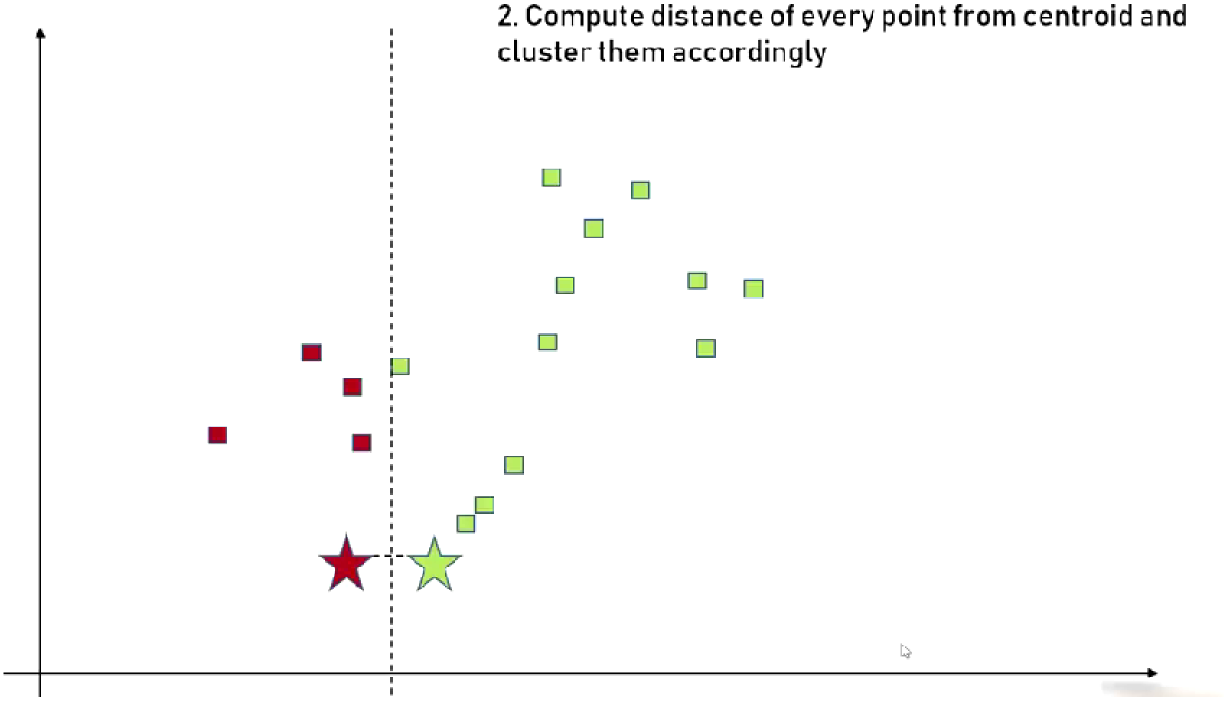
Clustering With K Means - Python Tutorial



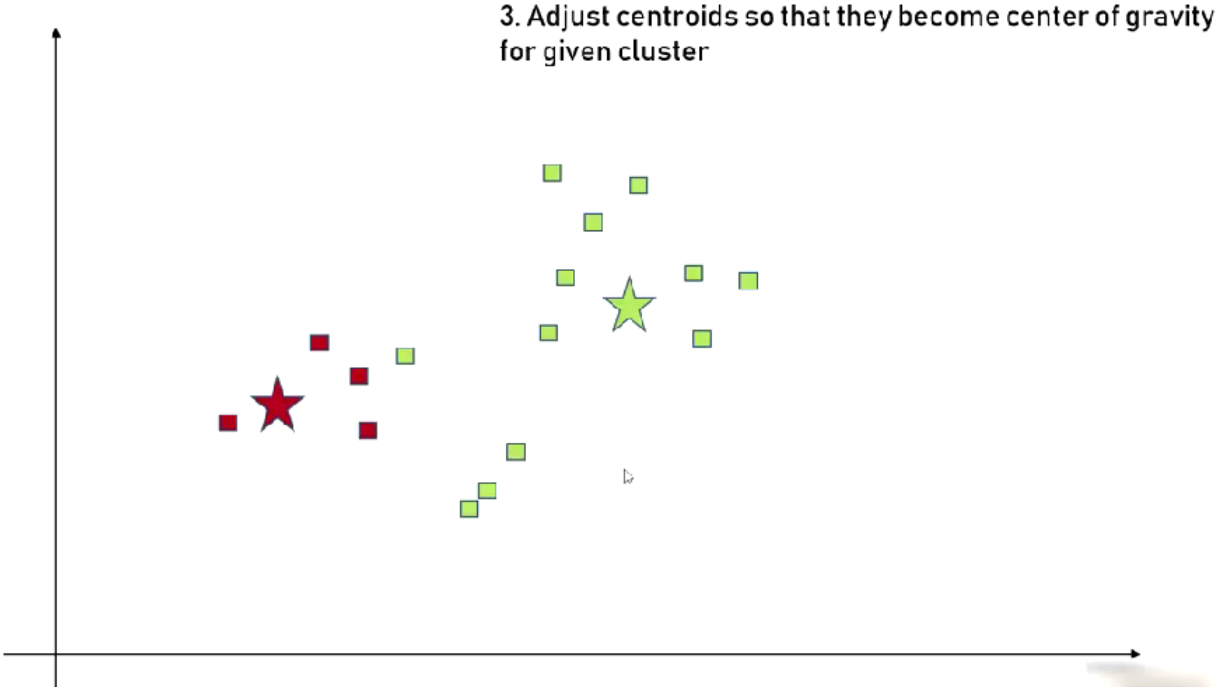


1. Start with K centroids by putting them at random place, Here k = 2

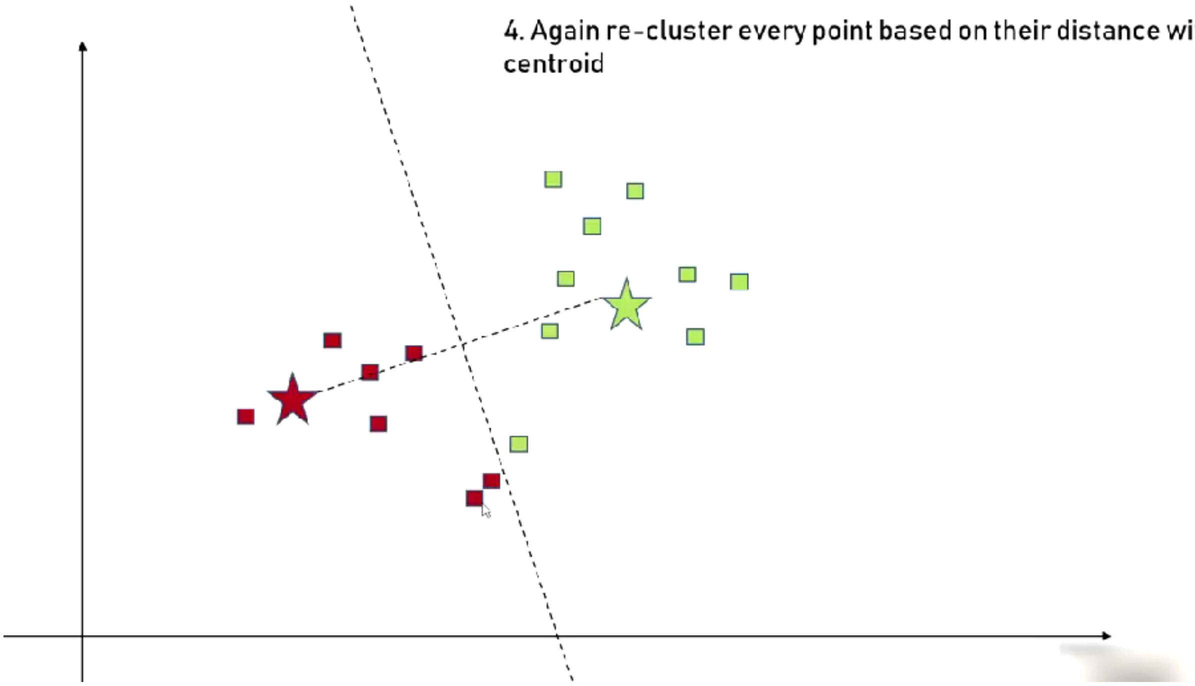
2. Compute distance of every point from centroid and cluster them accordingly



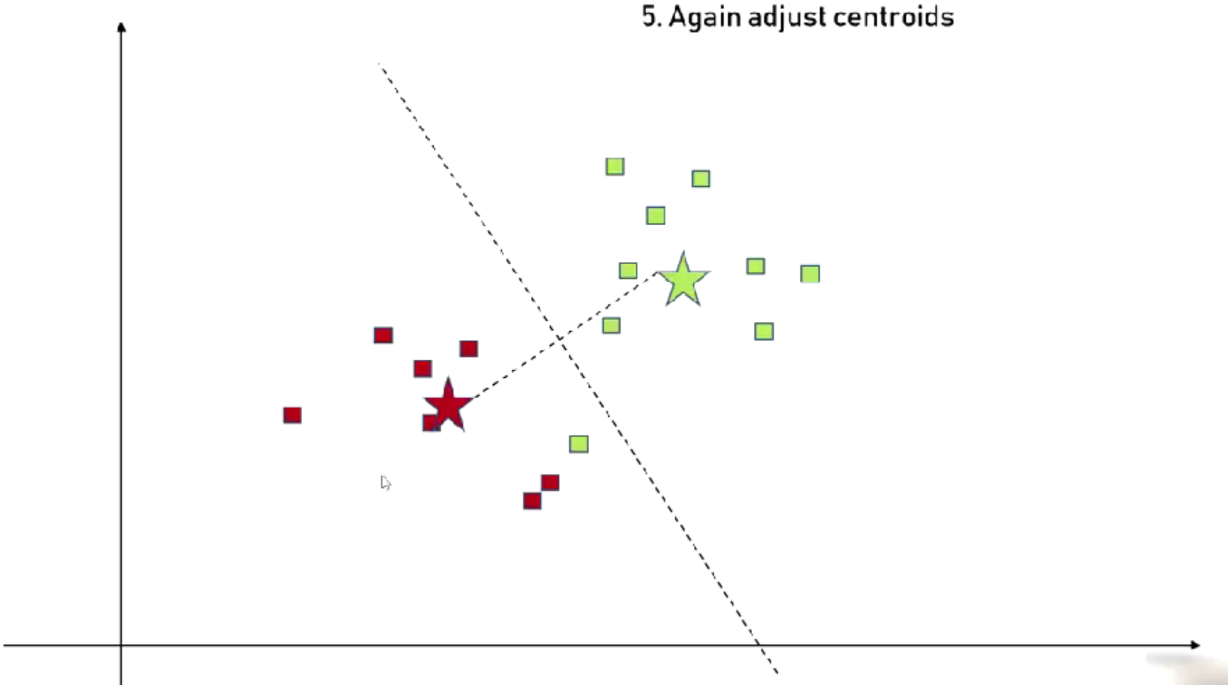
3. Adjust centroids so that they become center of gravity for given cluster

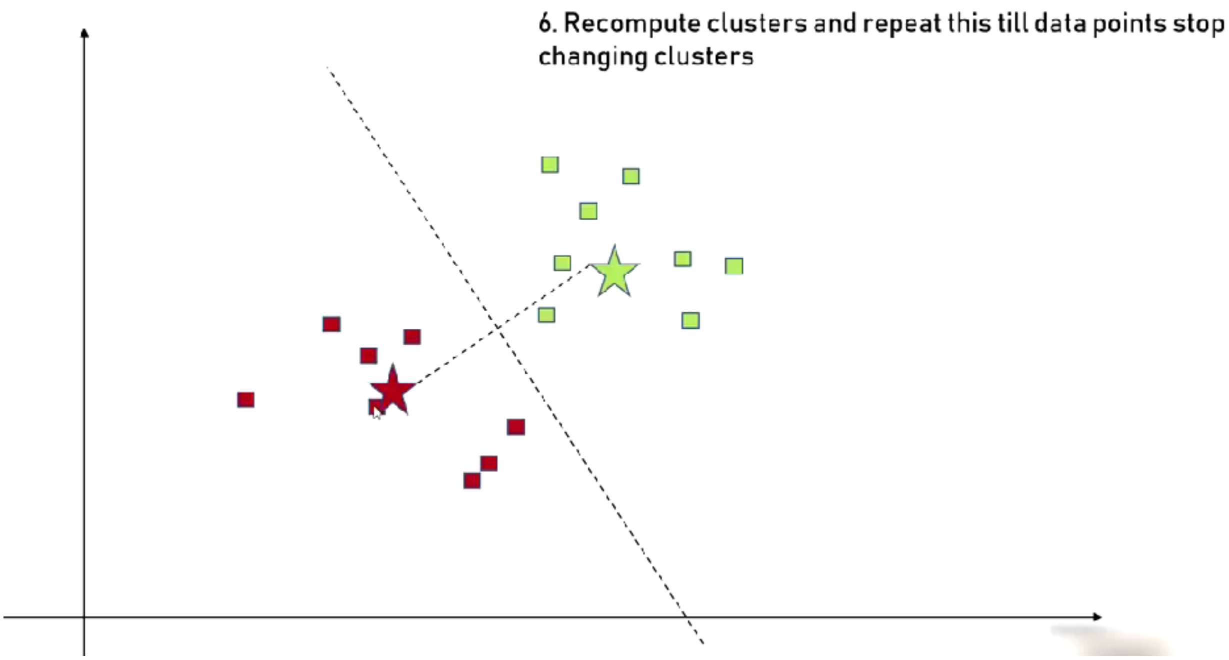


4. Again re-cluster every point based on their distance with centroid

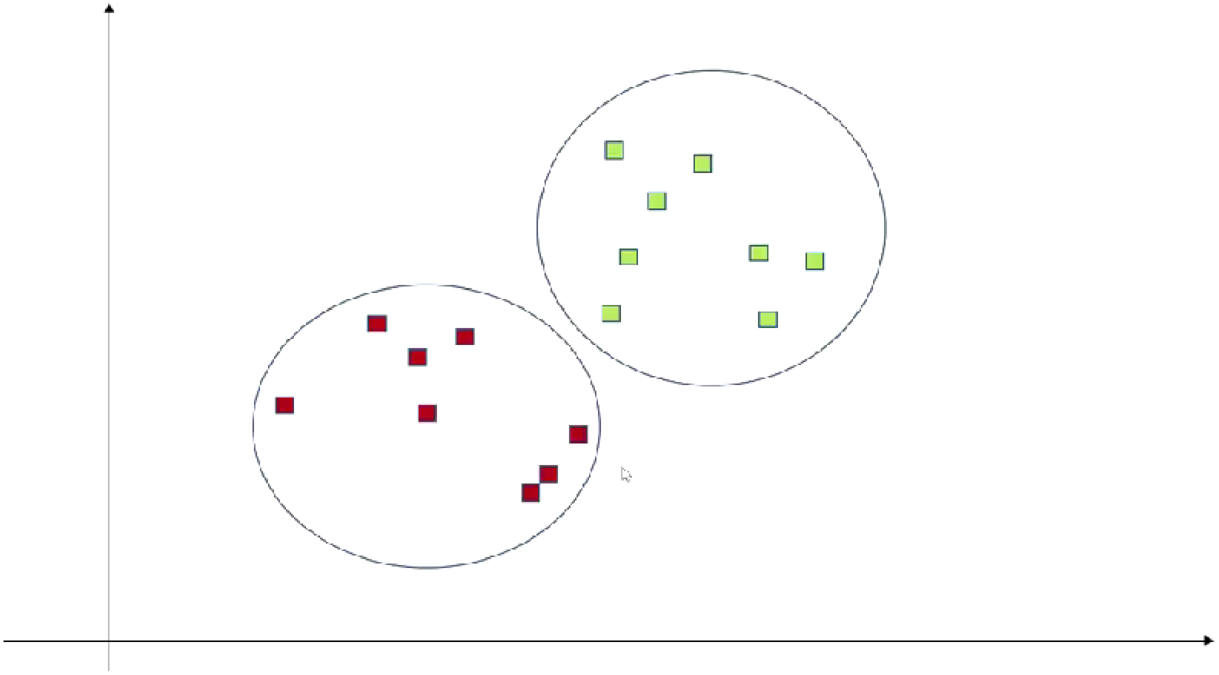


5. Again adjust centroids





6. Recompute clusters and repeat this till data points stop changing clusters



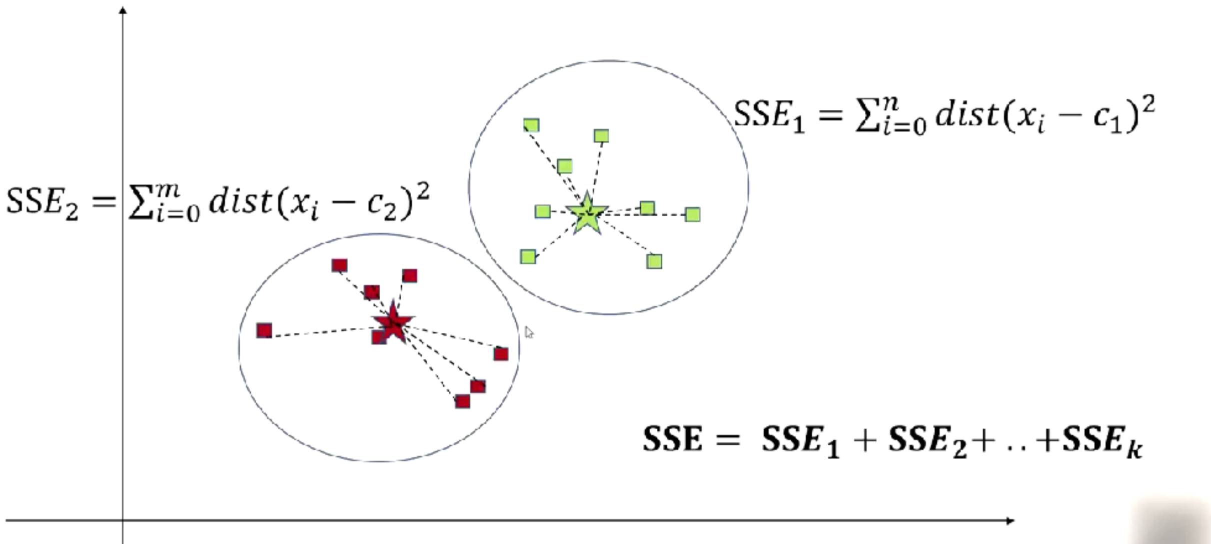
How to determine correct number of clusters (K)?

By using elbow method

Elbow method

You start with some k foe example k=2 & we tried to compute SSE for all clusters.

SSE = Sum of Squared Errors

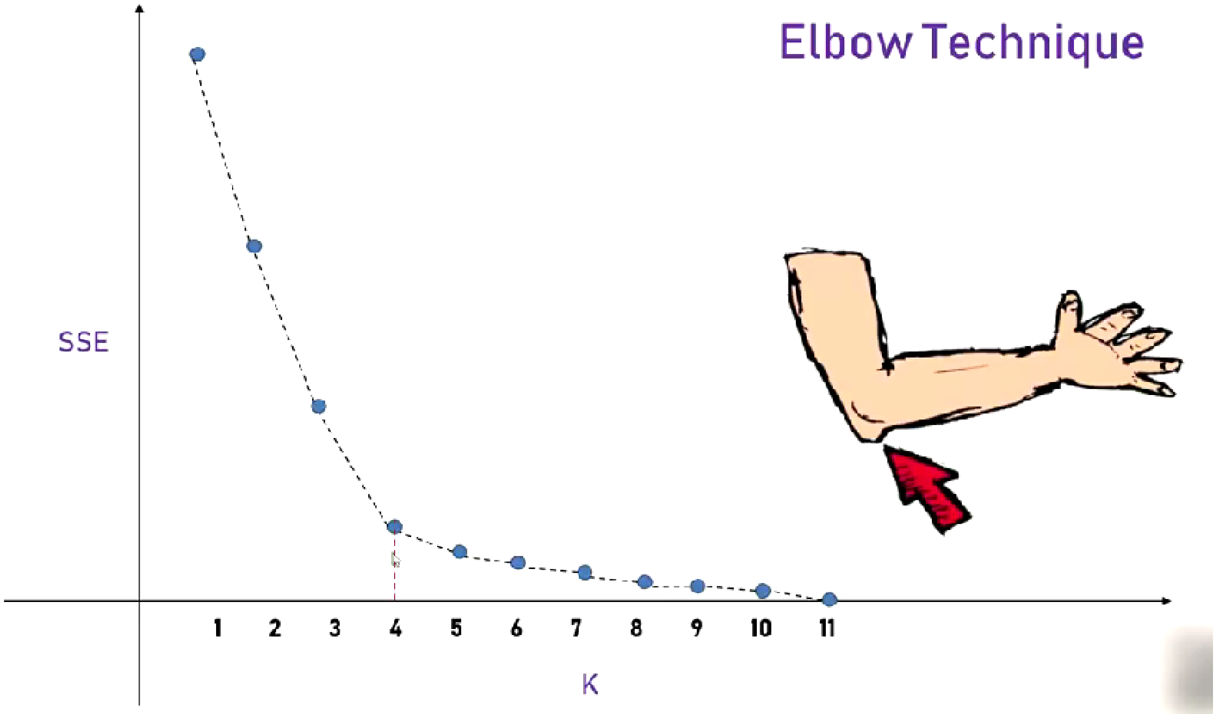


$SSE_1 = \sum_{i=0}^{n} dist(x_i - c_1)^2$

$SSE_2 = \sum_{i=0}^{m} dist(x_i - c_2)^2$

$\mathbf{SSE = SSE_1 + SSE_2 + .. + SSE_k}$

Once you have SSE you plot SSE versus k. You realize that as you increase the number of clusters, it will decrease the error. A general guidline is to find an elbow. Here is the good cluster number.



In [1]:

```
from sklearn.cluster import KMeans
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
from matplotlib import pyplot as plt
%matplotlib inline
```

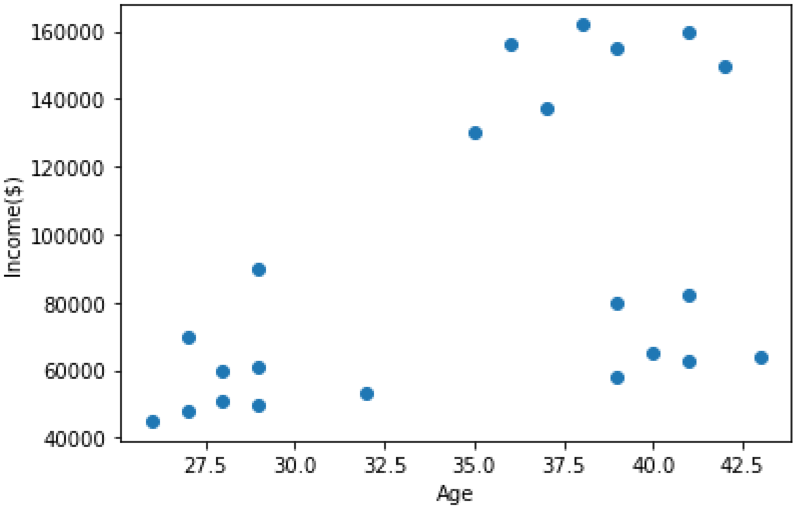
In [2]:
```
df = pd.read_csv("D:/Data_Science/My Github/Machine-Learning-with-Python/13. kmea
df.head()
```

Out[2]:

	Name	Age	Income($)
0	Rob	27	70000
1	Michael	29	90000
2	Mohan	29	61000
3	Ismail	28	60000
4	Kory	42	150000

In [3]:
```
plt.scatter(df.Age,df['Income($)'])
plt.xlabel('Age')
plt.ylabel('Income($)')
```

Out[3]:
```
Text(0, 0.5, 'Income($)')
```



In [4]:
```
km = KMeans(n_clusters=3)
y_predicted = km.fit_predict(df[['Age','Income($)']])
y_predicted
```

Out[4]:
```
array([2, 2, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 1])
```

In [5]:

```
#append new column
df['cluster']=y_predicted
df.head()
```

Out[5]:

	Name	Age	Income($)	cluster
0	Rob	27	70000	2
1	Michael	29	90000	2
2	Mohan	29	61000	1
3	Ismail	28	60000	1
4	Kory	42	150000	0

In [6]:

```
#centroids of clusters
km.cluster_centers_
```

Out[6]:

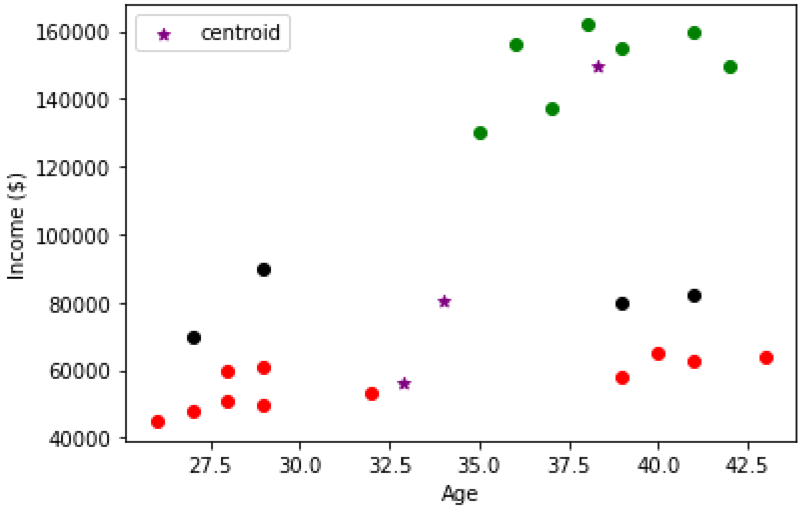
```
array([[3.82857143e+01, 1.50000000e+05],
       [3.29090909e+01, 5.61363636e+04],
       [3.40000000e+01, 8.05000000e+04]])
```

In [7]:

```
df1 = df[df.cluster==0]
df2 = df[df.cluster==1]
df3 = df[df.cluster==2]
plt.scatter(df1.Age,df1['Income($)'],color='green')
plt.scatter(df2.Age,df2['Income($)'],color='red')
plt.scatter(df3.Age,df3['Income($)'],color='black')
plt.scatter(km.cluster_centers_[:,0],km.cluster_centers_[:,1],color='purple',mark
plt.xlabel('Age')
plt.ylabel('Income ($)')
plt.legend()
```

Out[7]:

```
<matplotlib.legend.Legend at 0x5a90e20>
```



Preprocessing using min max scaler

In [8]:

```
scaler = MinMaxScaler()
scaler.fit(df[['Income($)']])
df['Income($)'] = scaler.transform(df[['Income($)']])
scaler.fit(df[['Age']])
df['Age'] = scaler.transform(df[['Age']])
```

In [9]:

```
df.head()
```

Out[9]:

	Name	Age	Income($)	cluster
0	Rob	0.058824	0.213675	2
1	Michael	0.176471	0.384615	2
2	Mohan	0.176471	0.136752	1
3	Ismail	0.117647	0.128205	1
4	Kory	0.941176	0.897436	0

In [10]:

```
km = KMeans(n_clusters=3)
y_predicted = km.fit_predict(df[['Age','Income($)']])
y_predicted
```

Out[10]:

```
array([0, 0, 0, 0, 2, 2, 2, 2, 2, 2, 2, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1])
```

In [11]:

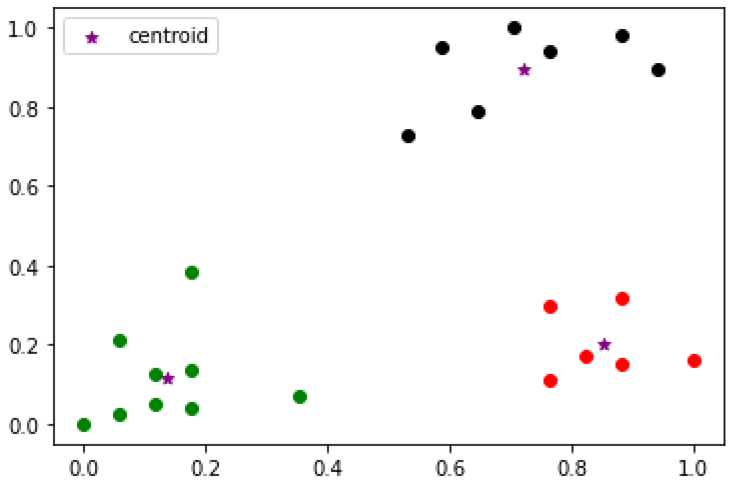
```
df['cluster']=y_predicted
df.head()
```

Out[11]:

	Name	Age	Income($)	cluster
0	Rob	0.058824	0.213675	0
1	Michael	0.176471	0.384615	0
2	Mohan	0.176471	0.136752	0
3	Ismail	0.117647	0.128205	0
4	Kory	0.941176	0.897436	2

In [12]:
```
df1 = df[df.cluster==0]
df2 = df[df.cluster==1]
df3 = df[df.cluster==2]
plt.scatter(df1.Age,df1['Income($)'],color='green')
plt.scatter(df2.Age,df2['Income($)'],color='red')
plt.scatter(df3.Age,df3['Income($)'],color='black')
plt.scatter(km.cluster_centers_[:,0],km.cluster_centers_[:,1],color='purple',mark
plt.legend()
```

Out[12]:
```
<matplotlib.legend.Legend at 0x5b17310>
```



Elbow Plot

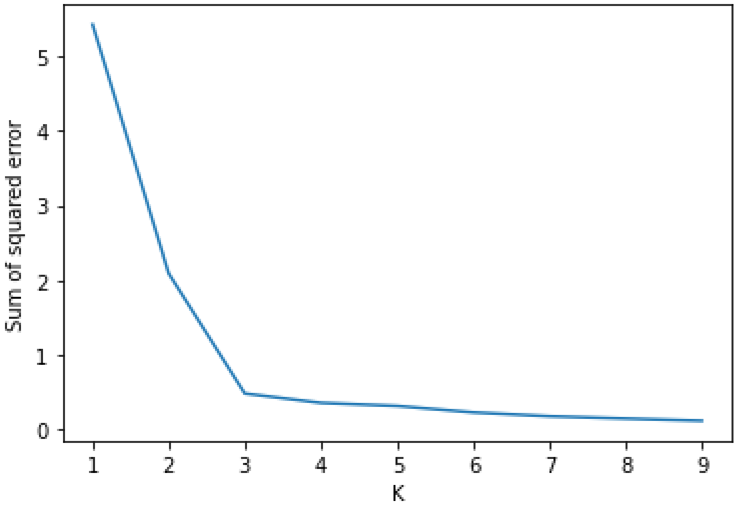
In [13]:
```
sse = []
k_rng = range(1,10)
for k in k_rng:
    km = KMeans(n_clusters=k)
    km.fit(df[['Age','Income($)']])
    sse.append(km.inertia_)
```

In [15]:
```
sse
```

Out[15]:
```
[5.434011511988179,
 2.091136388699078,
 0.4750783498553097,
 0.3491047094419566,
 0.30713504184752916,
 0.22020960864009395,
 0.1685851223602976,
 0.13781880133764024,
 0.10824862283029266]
```

In [16]:
```
plt.xlabel('K')
plt.ylabel('Sum of squared error')
plt.plot(k_rng,sse)
```

Out[16]:
```
[<matplotlib.lines.Line2D at 0xc772850>]
```



Date	Author
2021-10-04	**Ehsan Zia**