Practical 2 :

Aim :Implement K-means clustering

Data set : https://github.com/tonudon86/AI-practicals/blob/master/detasets/Income.csv

Theory :

k-means clustering is a method of vector quantization, originally from signal processing, that aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster.

Code :

In [28]:

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

In [29]:

```
df = pd.read_csv('./detasets/Income.csv')
df.head()
```

Out[29]:

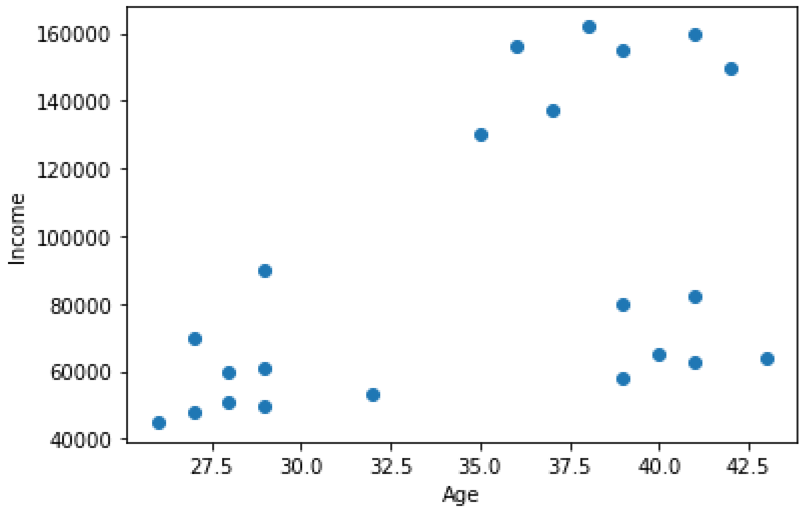
```
      Name     Age   Income
0      Rob      27    70000
1  Michael      29    90000
2    Mohan      29    61000
3   Ismail      28    60000
4     Kory      42   150000
```

In [30]:

```
plt.scatter(df.Age,df['Income'])
plt.xlabel('Age')
plt.ylabel('Income')
```

Out[30]:

```
Text(0, 0.5, 'Income')
```



Preproccing for scalling

In [31]:

```
scaler = MinMaxScaler()

scaler.fit(df[['Income']])
df['Income'] = scaler.transform(df[['Income']])

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

Elbo Method for Finding Best K Value

In [39]:

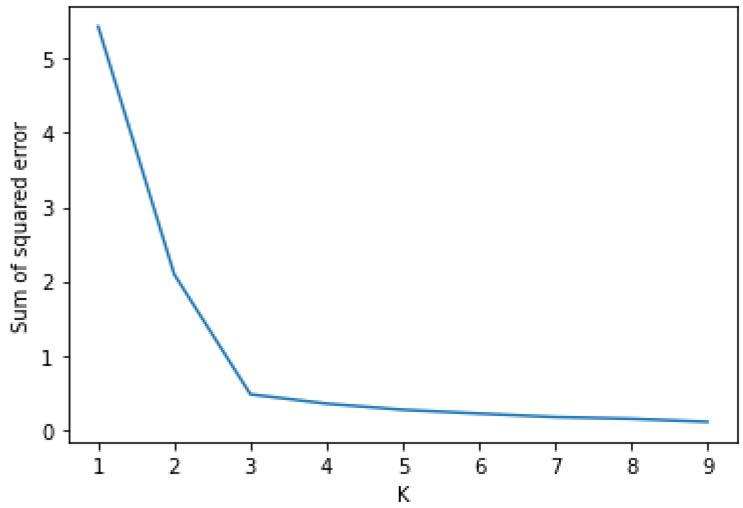
```
sse = []
k_rng = range(1,10)
for k in k_rng:
    km = KMeans(n_clusters=k)
    km.fit(df[['Age','Income']])
    sse.append(km.inertia_)
```

In [40]:

```
plt.xlabel('K')
plt.ylabel('Sum of squared error')
plt.plot(k_rng,sse)
```

Out[40]:

```
[<matplotlib.lines.Line2D at 0x7f7d5af50c10>]
```



with the help of graph we can select elbow is at k=3 so we can decide that the best value for k is 3

In [34]:

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

Out[34]:

```
array([1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2],
    dtype=int32)
```

In [23]:

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

Out[23]:

	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	1

In [24]:

```
## checking centroid of cluster
km.cluster_centers_
```

Out[24]:

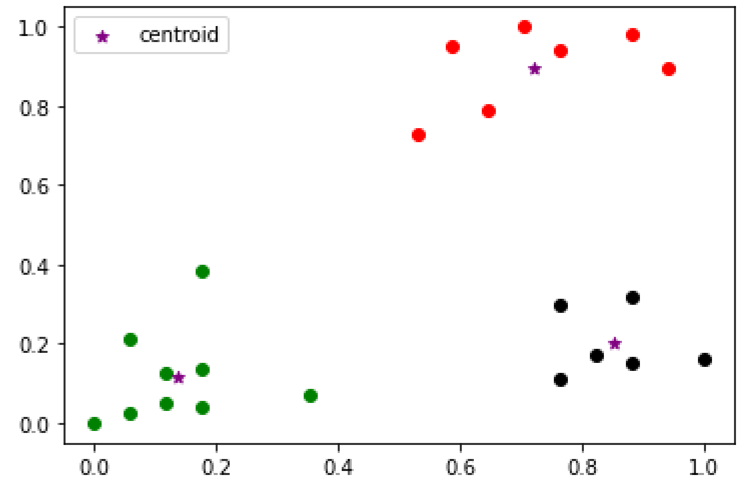
```
array([[0.1372549 , 0.11633428],
     [0.72268908, 0.8974359 ],
     [0.85294118, 0.2022792 ]])
```

In [26]:

```
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',marker='*',label='centroid')
plt.legend()
```

Out[26]:

```
<matplotlib.legend.Legend at 0x7f7d48c8c970>
```



In [ ]:

Code link :

https://github.com/tonudon86/AI-practicals/blob/master/Practical2.ipynb