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

In [2]: df=pd.read\_csv("em.csv")

In [3]: df

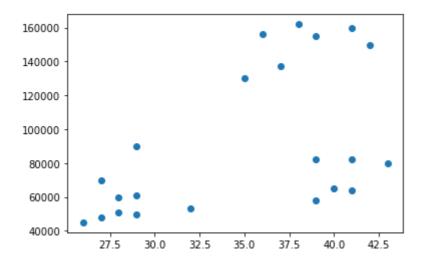
#### Out[3]:

	name	age	income
0	Hritik	27	70000
1	Arpit	29	90000
2	Manav	29	61000
3	Kirti	28	60000
4	Siddhi	42	150000
5	Riya	39	155000
6	Ankita	41	160000
7	Vikash	38	162000
8	Priyank	36	156000
9	Kranti	35	130000
10	Rohit	37	137000
11	Aakash	26	45000
12	Durgesh	27	48000
13	Varun	28	51000
14	Vicky	29	49500
15	Priyank	32	53000
16	Kranti	40	65000
17	Rohit	41	64000
18	Aakash	43	80000
19	Durgesh	39	82000
20	Varun	41	82000
21	Vicky	39	58000

In [4]: from matplotlib import pyplot as plt

```
In [5]: plt.scatter(df['age'],df['income'])
```

Out[5]: <matplotlib.collections.PathCollection at 0x1efdcab31f0>



### **Apply K-means clustering**

```
In [6]: from sklearn.cluster import KMeans
In [7]: kmean=KMeans(n_clusters=3)
In [8]: kmean
Out[8]: KMeans(n_clusters=3)
```

### Assign the clusters to the data

```
In [9]: y_predict=kmean.fit_predict(df[['age','income']])
In [10]: y_predict
Out[10]: array([0, 0, 2, 2, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 0, 0, 0, 2])
```

#### Revise the dataframe with cluster name

```
In [11]: df['cluster']=y_predict
```

In [12]: d

Out[12]:

	name	age	income	cluster
0	Hritik	27	70000	0
1	Arpit	29	90000	0
2	Manav	29	61000	2
3	Kirti	28	60000	2
4	Siddhi	42	150000	1
5	Riya	39	155000	1
6	Ankita	41	160000	1
7	Vikash	38	162000	1
8	Priyank	36	156000	1
9	Kranti	35	130000	1
10	Rohit	37	137000	1
11	Aakash	26	45000	2
12	Durgesh	27	48000	2
13	Varun	28	51000	2
14	Vicky	29	49500	2
15	Priyank	32	53000	2
16	Kranti	40	65000	2
17	Rohit	41	64000	2
18	Aakash	43	80000	0
19	Durgesh	39	82000	0
20	Varun	41	82000	0
21	Vicky	39	58000	2

## predict the cluster of given data

name: ABC age:34 income:50000

```
In [13]: kmean.predict([[34,500000]])

C:\Users\Hiray\anaconda3\lib\site-packages\sklearn\base.py:450: UserWarning:
    X does not have valid feature names, but KMeans was fitted with feature name
    s
    warnings.warn(
```

Out[13]: array([1])

# divide the dataframe according to the clusters

```
In [14]:
         df1=df[df.cluster==0]
         df2=df[df.cluster==1]
         df3=df[df.cluster==2]
         plt.scatter(df1.age,df1.income,color='blue')
In [15]:
         plt.scatter(df2.age,df2.income,color='green')
         plt.scatter(df3.age,df3.income,color='red')
         plt.xlabel('age')
         plt.ylabel('income')
Out[15]: Text(0, 0.5, 'income')
            160000
            140000
            120000
            100000
             80000
             60000
```

# Scale the attribute values from 0 to 1 by applying min-max scaler technique

35.0 age

30.0

37.5

42.5

```
In [16]: from sklearn.preprocessing import MinMaxScaler
In [17]: scaler=MinMaxScaler()
In [18]: scaler.fit(df[['age']])
Out[18]: MinMaxScaler()
In [19]: df['AGE']=scaler.transform(df[['age']])
```

40000

In [20]: df

Out[20]:

	name	age	income	cluster	AGE
0	Hritik	27	70000	0	0.058824
1	Arpit	29	90000	0	0.176471
2	Manav	29	61000	2	0.176471
3	Kirti	28	60000	2	0.117647
4	Siddhi	42	150000	1	0.941176
5	Riya	39	155000	1	0.764706
6	Ankita	41	160000	1	0.882353
7	Vikash	38	162000	1	0.705882
8	Priyank	36	156000	1	0.588235
9	Kranti	35	130000	1	0.529412
10	Rohit	37	137000	1	0.647059
11	Aakash	26	45000	2	0.000000
12	Durgesh	27	48000	2	0.058824
13	Varun	28	51000	2	0.117647
14	Vicky	29	49500	2	0.176471
15	Priyank	32	53000	2	0.352941
16	Kranti	40	65000	2	0.823529
17	Rohit	41	64000	2	0.882353
18	Aakash	43	80000	0	1.000000
19	Durgesh	39	82000	0	0.764706
20	Varun	41	82000	0	0.882353
21	Vicky	39	58000	2	0.764706

```
In [21]: scaler.fit(df[['income']])
```

Out[21]: MinMaxScaler()

```
In [22]: df['INCOME']=scaler.transform(df[['income']])
```

In [23]: df

Out[23]:

	name	age	income	cluster	AGE	INCOME
0	Hritik	27	70000	0	0.058824	0.213675
1	Arpit	29	90000	0	0.176471	0.384615
2	Manav	29	61000	2	0.176471	0.136752
3	Kirti	28	60000	2	0.117647	0.128205
4	Siddhi	42	150000	1	0.941176	0.897436
5	Riya	39	155000	1	0.764706	0.940171
6	Ankita	41	160000	1	0.882353	0.982906
7	Vikash	38	162000	1	0.705882	1.000000
8	Priyank	36	156000	1	0.588235	0.948718
9	Kranti	35	130000	1	0.529412	0.726496
10	Rohit	37	137000	1	0.647059	0.786325
11	Aakash	26	45000	2	0.000000	0.000000
12	Durgesh	27	48000	2	0.058824	0.025641
13	Varun	28	51000	2	0.117647	0.051282
14	Vicky	29	49500	2	0.176471	0.038462
15	Priyank	32	53000	2	0.352941	0.068376
16	Kranti	40	65000	2	0.823529	0.170940
17	Rohit	41	64000	2	0.882353	0.162393
18	Aakash	43	80000	0	1.000000	0.299145
19	Durgesh	39	82000	0	0.764706	0.316239
20	Varun	41	82000	0	0.882353	0.316239
21	Vicky	39	58000	2	0.764706	0.111111

# Again apply k-means clustering on the scaled attribute

```
In [24]: km=KMeans(n_clusters=3)
    y_predicted=km.fit_predict(df[['AGE','INCOME']])

In [25]: y_predicted

Out[25]: array([0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 2, 2, 2, 2, 2])

In [26]: df['cluster']=y_predicted
```

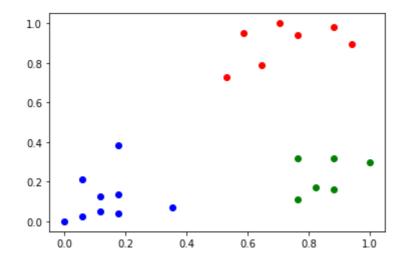
In [27]: df

Out[27]:

name	age	income	cluster	AGE	INCOME
Hritik	27	70000	0	0.058824	0.213675
Arpit	29	90000	0	0.176471	0.384615
Manav	29	61000	0	0.176471	0.136752
Kirti	28	60000	0	0.117647	0.128205
Siddhi	42	150000	1	0.941176	0.897436
Riya	39	155000	1	0.764706	0.940171
Ankita	41	160000	1	0.882353	0.982906
Vikash	38	162000	1	0.705882	1.000000
Priyank	36	156000	1	0.588235	0.948718
Kranti	35	130000	1	0.529412	0.726496
Rohit	37	137000	1	0.647059	0.786325
Aakash	26	45000	0	0.000000	0.000000
Durgesh	27	48000	0	0.058824	0.025641
Varun	28	51000	0	0.117647	0.051282
Vicky	29	49500	0	0.176471	0.038462
Priyank	32	53000	0	0.352941	0.068376
Kranti	40	65000	2	0.823529	0.170940
Rohit	41	64000	2	0.882353	0.162393
Aakash	43	80000	2	1.000000	0.299145
Durgesh	39	82000	2	0.764706	0.316239
Varun	41	82000	2	0.882353	0.316239
Vicky	39	58000	2	0.764706	0.111111
	Hritik Arpit Manav Kirti Siddhi Riya Ankita Vikash Priyank Kranti Rohit Aakash Durgesh Varun Vicky Priyank Kranti Rohit Aohit	Hritik 27 Arpit 29 Manav 29 Kirti 28 Siddhi 42 Riya 39 Ankita 41 Vikash 38 Priyank 36 Kranti 35 Rohit 37 Aakash 26 Durgesh 27 Varun 28 Vicky 29 Priyank 32 Kranti 40 Rohit 41 Aakash 43 Durgesh 39 Varun 41	Hritik         27         70000           Arpit         29         90000           Manav         29         61000           Kirti         28         60000           Siddhi         42         150000           Riya         39         155000           Ankita         41         160000           Vikash         38         162000           Priyank         36         156000           Kranti         37         137000           Aakash         26         45000           Durgesh         27         48000           Varun         28         51000           Vicky         29         49500           Priyank         32         53000           Kranti         40         65000           Rohit         41         64000           Aakash         43         80000           Durgesh         39         82000           Varun         41         82000	Hritik 27 70000 0 Arpit 29 90000 0 Manav 29 61000 0 Kirti 28 60000 1 Siddhi 42 150000 1 Riya 39 155000 1 Ankita 41 160000 1 Vikash 38 162000 1 Priyank 36 156000 1 Kranti 35 130000 1 Rohit 37 137000 1 Aakash 26 45000 0 Durgesh 27 48000 0 Varun 28 51000 0 Vicky 29 49500 0 Priyank 32 53000 0 Kranti 40 65000 2 Rohit 41 64000 2 Aakash 43 80000 2 Durgesh 39 82000 2 Varun 41 82000 2	Hritik 27 70000 0 0.058824 Arpit 29 90000 0 0.176471 Manav 29 61000 0 0.176471 Kirti 28 60000 0 0.117647 Siddhi 42 150000 1 0.941176 Riya 39 155000 1 0.764706 Ankita 41 160000 1 0.882353 Vikash 38 162000 1 0.705882 Priyank 36 156000 1 0.588235 Kranti 35 130000 1 0.529412 Rohit 37 137000 1 0.647059 Aakash 26 45000 0 0.000000 Durgesh 27 48000 0 0.058824 Varun 28 51000 0 0.117647 Vicky 29 49500 0 0.117647 Vicky 29 49500 0 0.176471 Priyank 32 53000 0 0.352941 Kranti 40 65000 2 0.823529 Rohit 41 64000 2 0.882353 Aakash 43 80000 2 1.000000 Durgesh 39 82000 2 0.764706 Varun 41 82000 2 0.882353

```
In [28]: df1=df[df.cluster==0]
    df2=df[df.cluster==1]
    df3=df[df.cluster==2]
    plt.scatter(df1.AGE,df1.INCOME,color='blue')
    plt.scatter(df2.AGE,df2.INCOME,color='red')
    plt.scatter(df3.AGE,df3.INCOME,color='green')
```

Out[28]: <matplotlib.collections.PathCollection at 0x1efdf5b1250>



### **Display cluster means**

### **Assignment**

## Implement K-means clustering on the following dataset

object	weight	PH
Medicine A	1	1
Medicine B	2	1
Medicine C	4	3
Medicine D	5	4

#### **Elbow Method**

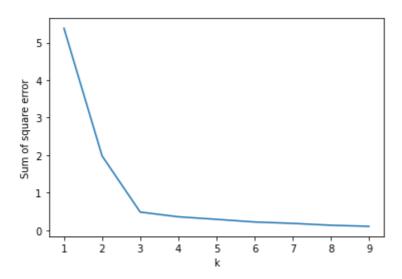
```
In [30]:
         k_range=range(1,10)
         sse=[]
         for k in k_range:
             km = KMeans(n_clusters=k)
             km.fit(df[['AGE','INCOME']])
                                                            #fit is used for train the mo
             sse.append(km.inertia_)
                                                               #inertia is
```

C:\Users\Hiray\anaconda3\lib\site-packages\sklearn\cluster\\_kmeans.py:1036: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting th e environment variable OMP\_NUM\_THREADS=1.

warnings.warn(

```
In [31]:
Out[31]: [5.383034948191102,
          1.9781765163703693,
          0.48132424071658514,
          0.35535060030323207,
          0.2880938652978993,
          0.21680068081600046,
          0.1798018564144019,
          0.13008339301480998,
          0.10126672060839395]
         plt.xlabel('k')
In [32]:
         plt.ylabel("Sum of square error")
         plt.plot(k_range, sse)
```

#### Out[32]: [<matplotlib.lines.Line2D at 0x1efdf633580>]



```
iris = pd.read_csv("Iris.csv")
In [33]:
```

In [34]: iris

Out[34]:

	ld	SepalLength	SepalWidth	PetalLength	PetalWidth	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa
145	146	6.7	3.0	5.2	2.3	Iris-virginica
146	147	6.3	2.5	5.0	1.9	Iris-virginica
147	148	6.5	3.0	5.2	2.0	Iris-virginica
148	149	6.2	3.4	5.4	2.3	Iris-virginica
149	150	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 6 columns

```
In [38]: iris.shape
```

Out[38]: (150, 6)

```
In [ ]: kmi = KMeans(n_clusters=3)
y_predict = kmi.fit_predict(iris[['age','income']])
```