Problem Statement:

```
For this model we will implement k-Means Clustering
```

In [1]:

```
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
```

In [2]:
```
df=pd.read_csv(r"C:\Users\my pc\Downloads\Online Retail.csv")
df
```

Out[2]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	01-12-2010 08:26	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	01-12-2010 08:26	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	01-12-2010 08:26	2.75	17850.0	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	01-12-2010 08:26	3.39	17850.0	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	01-12-2010 08:26	3.39	17850.0	United Kingdom
...	...	...	...	...	...	...	...	...
541904	581587	22613	PACK OF 20 SPACEBOY NAPKINS	12	09-12-2011 12:50	0.85	12680.0	France
541905	581587	22899	CHILDREN'S APRON DOLLY GIRL	6	09-12-2011 12:50	2.10	12680.0	France
541906	581587	23254	CHILDRENS CUTLERY DOLLY GIRL	4	09-12-2011 12:50	4.15	12680.0	France
541907	581587	23255	CHILDRENS CUTLERY CIRCUS PARADE	4	09-12-2011 12:50	4.15	12680.0	France
541908	581587	22138	BAKING SET 9 PIECE RETROSPOT	3	09-12-2011 12:50	4.95	12680.0	France

541909 rows × 8 columns

Data cleaning and Preprocessing

In [3]:

```
df.head()
```

Out[3]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	01-12-2010 08:26	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	01-12-2010 08:26	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	01-12-2010 08:26	2.75	17850.0	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	01-12-2010 08:26	3.39	17850.0	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	01-12-2010 08:26	3.39	17850.0	United Kingdom

In [4]:

```
df.tail()
```

Out[4]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
541904	581587	22613	PACK OF 20 SPACEBOY NAPKINS	12	09-12-2011 12:50	0.85	12680.0	France
541905	581587	22899	CHILDREN'S APRON DOLLY GIRL	6	09-12-2011 12:50	2.10	12680.0	France
541906	581587	23254	CHILDRENS CUTLERY DOLLY GIRL	4	09-12-2011 12:50	4.15	12680.0	France
541907	581587	23255	CHILDRENS CUTLERY CIRCUS PARADE	4	09-12-2011 12:50	4.15	12680.0	France
541908	581587	22138	BAKING SET 9 PIECE RETROSPOT	3	09-12-2011 12:50	4.95	12680.0	France

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 541909 entries, 0 to 541908
Data columns (total 8 columns):
 #   Column       Non-Null Count   Dtype  
---  ------       --------------   -----  
 0   InvoiceNo    541909 non-null  object 
 1   StockCode    541909 non-null  object 
 2   Description  540455 non-null  object 
 3   Quantity     541909 non-null  int64  
 4   InvoiceDate  541909 non-null  object 
 5   UnitPrice    541909 non-null  float64
 6   CustomerID   406829 non-null  float64
 7   Country      541909 non-null  object 
dtypes: float64(2), int64(1), object(5)
memory usage: 33.1+ MB
```

In [6]:
```
df.describe()
```

Out[6]:

	Quantity	UnitPrice	CustomerID
count	541909.000000	541909.000000	406829.000000
mean	9.552250	4.611114	15287.690570
std	218.081158	96.759853	1713.600303
min	-80995.000000	-11062.060000	12346.000000
25%	1.000000	1.250000	13953.000000
50%	3.000000	2.080000	15152.000000
75%	10.000000	4.130000	16791.000000
max	80995.000000	38970.000000	18287.000000

In [7]:
```
df.shape
```

Out[7]:
```
(541909, 8)
```

In [8]:
```
df.isna().any()
```

Out[8]:
```
InvoiceNo      False
StockCode      False
Description     True
Quantity       False
InvoiceDate    False
UnitPrice      False
CustomerID      True
Country        False
dtype: bool
```

In [9]:
```
df["InvoiceNo"].value_counts()
```

Out[9]:
```
InvoiceNo
573585     1114
581219      749
581492      731
580729      721
558475      705
           ... 
554023        1
554022        1
554021        1
554020        1
C558901       1
Name: count, Length: 25900, dtype: int64
```

In [10]:
```
df=df.drop(["InvoiceNo","StockCode","Description","InvoiceDate"],axis=1)
```

In [11]:

```
df
```

Out[11]:

	Quantity	UnitPrice	CustomerID	Country
0	6	2.55	17850.0	United Kingdom
1	6	3.39	17850.0	United Kingdom
2	8	2.75	17850.0	United Kingdom
3	6	3.39	17850.0	United Kingdom
4	6	3.39	17850.0	United Kingdom
...	...	...	...	...
541904	12	0.85	12680.0	France
541905	6	2.10	12680.0	France
541906	4	4.15	12680.0	France
541907	4	4.15	12680.0	France
541908	3	4.95	12680.0	France

541909 rows × 4 columns

In [12]:

```
df.fillna(method="ffill",inplace=True)  #filling the null values using forward fill method
```

In [13]:

```
df.isna().any()
```

Out[13]:

```
Quantity      False
UnitPrice     False
CustomerID    False
Country       False
dtype: bool
```

```
In [14]: df.isnull().sum()
```

```
Out[14]: Quantity      0
         UnitPrice     0
         CustomerID    0
         Country       0
         dtype: int64
```

```
In [15]: df
```

Out[15]:

	Quantity	UnitPrice	CustomerID	Country
0	6	2.55	17850.0	United Kingdom
1	6	3.39	17850.0	United Kingdom
2	8	2.75	17850.0	United Kingdom
3	6	3.39	17850.0	United Kingdom
4	6	3.39	17850.0	United Kingdom
...	...	...	...	...
541904	12	0.85	12680.0	France
541905	6	2.10	12680.0	France
541906	4	4.15	12680.0	France
541907	4	4.15	12680.0	France
541908	3	4.95	12680.0	France

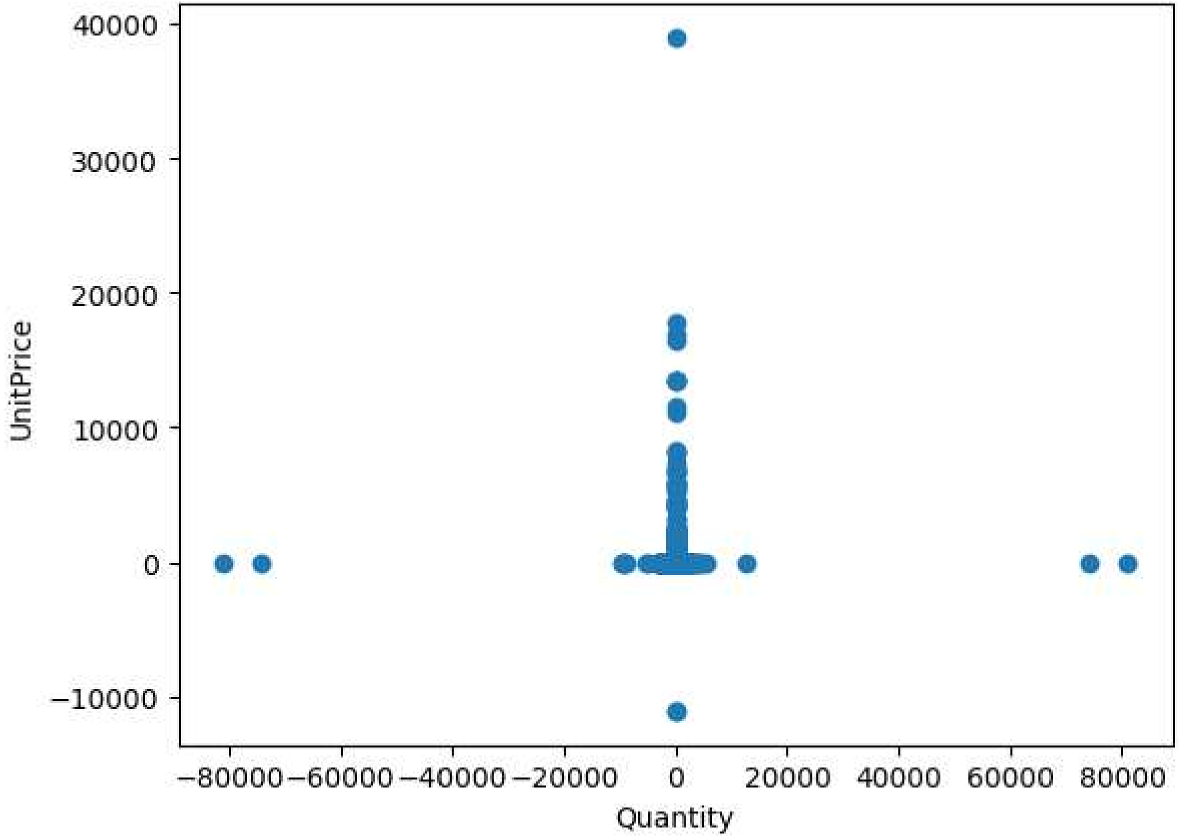
541909 rows × 4 columns

In [16]:
```
df["Country"].value_counts()
```

Out[16]:
```
Country
United Kingdom          495478
Germany                   9495
France                    8557
EIRE                      8196
Spain                     2533
Netherlands               2371
Belgium                   2069
Switzerland               2002
Portugal                  1519
Australia                 1259
Norway                    1086
Italy                      803
Channel Islands            758
Finland                    695
Cyprus                     622
Sweden                     462
Unspecified                446
Austria                    401
Denmark                    389
Japan                      358
Poland                     341
Israel                     297
USA                        291
Hong Kong                  288
Singapore                  229
Iceland                    182
Canada                     151
Greece                     146
Malta                      127
United Arab Emirates        68
European Community          61
RSA                         58
Lebanon                     45
Lithuania                   35
Brazil                      32
Czech Republic              30
Bahrain                     19
Saudi Arabia                10
Name: count, dtype: int64
```

In [17]:
```
plt.scatter(df["Quantity"],df["UnitPrice"])
plt.xlabel("Quantity")
plt.ylabel("UnitPrice")
```

Out[17]:
```
Text(0, 0.5, 'UnitPrice')
```



In [18]:
```
from sklearn.cluster import KMeans
km=KMeans()
km
```

Out[18]:
```
▾ KMeans

KMeans()
```

In [19]:
```
y_predicted=km.fit_predict(df[["Quantity","UnitPrice"]])
y_predicted
```

```
C:\Users\my pc\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\cluster\_kmeans.py:870: Fut
ureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` ex
plicitly to suppress the warning
  warnings.warn(
```

Out[19]:
```
array([0, 0, 0, ..., 0, 0, 0])
```

In [20]:
```
df["cluster"]=y_predicted
```

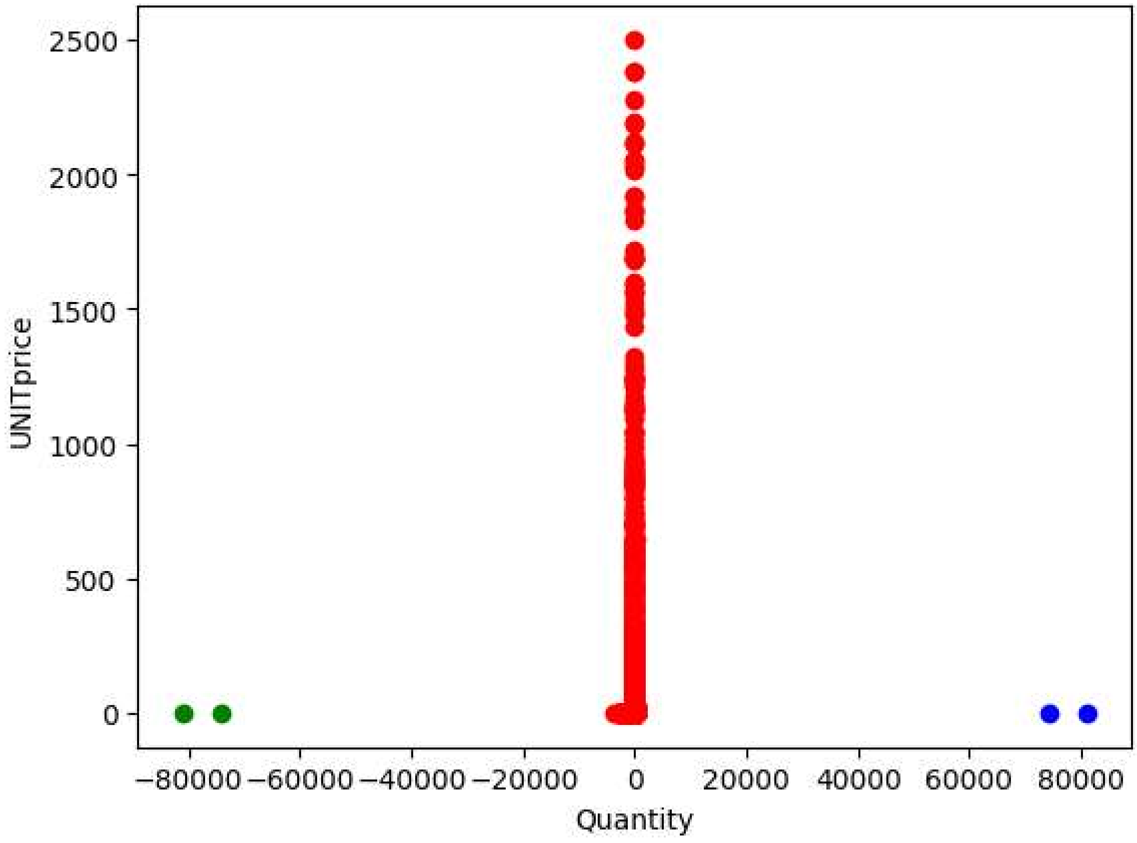
In [21]:
```
df.head()
```

Out[21]:

	Quantity	UnitPrice	CustomerID	Country	cluster
0	6	2.55	17850.0	United Kingdom	0
1	6	3.39	17850.0	United Kingdom	0
2	8	2.75	17850.0	United Kingdom	0
3	6	3.39	17850.0	United Kingdom	0
4	6	3.39	17850.0	United Kingdom	0

In [22]:
```
df1=df[df.cluster==0]
df2=df[df.cluster==1]
df3=df[df.cluster==2]
plt.scatter(df1["Quantity"],df1["UnitPrice"],color="red")
plt.scatter(df2["Quantity"],df2["UnitPrice"],color="green")
plt.scatter(df3["Quantity"],df3["UnitPrice"],color="blue")
plt.xlabel("Quantity")
plt.ylabel("UNITprice")
```

Out[22]:
```
Text(0, 0.5, 'UNITprice')
```



In [23]:
```
from sklearn.preprocessing import MinMaxScaler
```

In [24]:
```
Scaler=MinMaxScaler()
```

In [25]:
```
Scaler.fit(df[["Quantity"]])
```

Out[25]:
```
▾ MinMaxScaler
MinMaxScaler()
```

In [26]:
```
df["Quantity"]=Scaler.transform(df[["Quantity"]])
df.head()
```

Out[26]:

	Quantity	UnitPrice	CustomerID	Country	cluster
0	0.500037	2.55	17850.0	United Kingdom	0
1	0.500037	3.39	17850.0	United Kingdom	0
2	0.500049	2.75	17850.0	United Kingdom	0
3	0.500037	3.39	17850.0	United Kingdom	0
4	0.500037	3.39	17850.0	United Kingdom	0

In [27]:
```
Scaler.fit(df[["UnitPrice"]])
```

Out[27]:
```
▾ MinMaxScaler
MinMaxScaler()
```

In [28]:

```
df["UnitPrice"]=Scaler.transform(df[["UnitPrice"]])
df.head()
```

Out[28]:

	Quantity	UnitPrice	CustomerID	Country	cluster
0	0.500037	0.221150	17850.0	United Kingdom	0
1	0.500037	0.221167	17850.0	United Kingdom	0
2	0.500049	0.221154	17850.0	United Kingdom	0
3	0.500037	0.221167	17850.0	United Kingdom	0
4	0.500037	0.221167	17850.0	United Kingdom	0

In [29]:

```
km=KMeans()
```

In [30]:

```
y_predicted=km.fit_predict(df[["Quantity","UnitPrice"]])
y_predicted
```

```
C:\Users\my pc\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\cluster\_kmeans.py:870: Fut
ureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` ex
plicitly to suppress the warning
  warnings.warn(
```

Out[30]:

```
array([0, 0, 0, ..., 0, 0, 0])
```

In [31]:

```
df["New Cluster"]=y_predicted
df.head()
```

Out[31]:

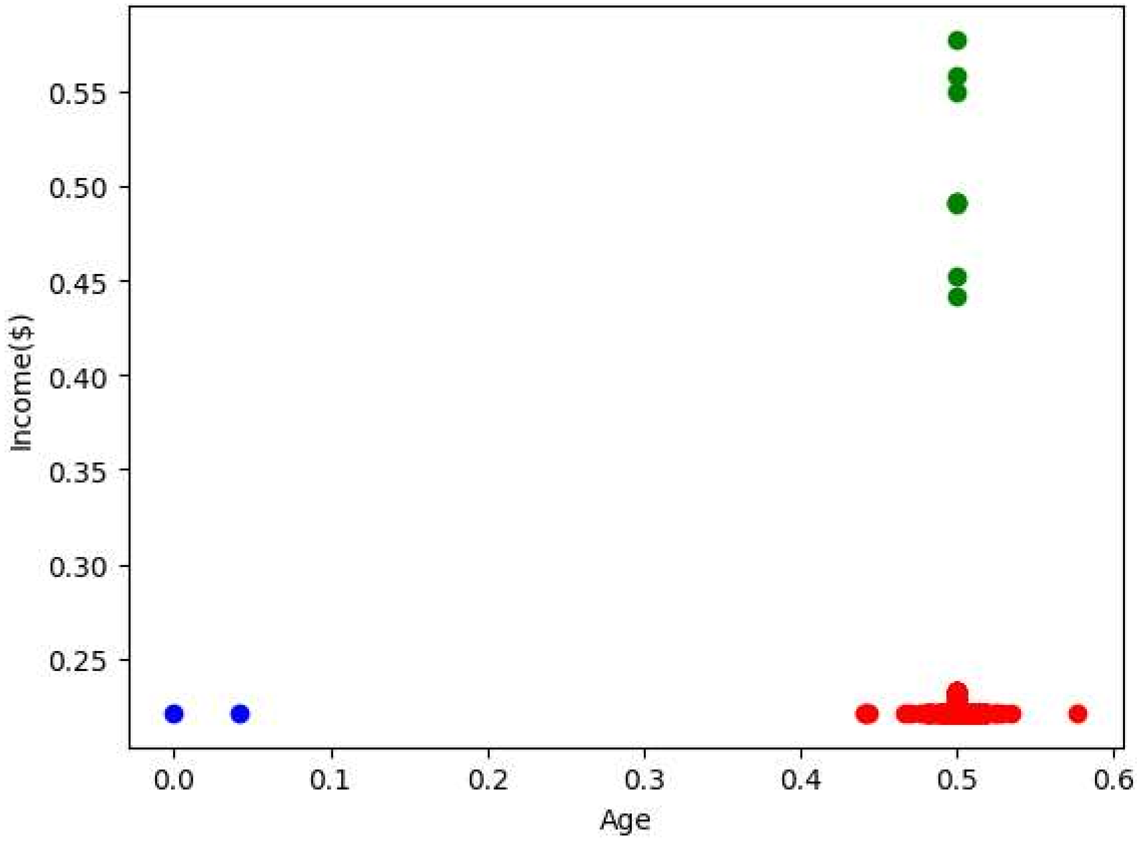
	Quantity	UnitPrice	CustomerID	Country	cluster	New Cluster
0	0.500037	0.221150	17850.0	United Kingdom	0	0
1	0.500037	0.221167	17850.0	United Kingdom	0	0
2	0.500049	0.221154	17850.0	United Kingdom	0	0
3	0.500037	0.221167	17850.0	United Kingdom	0	0
4	0.500037	0.221167	17850.0	United Kingdom	0	0

In [32]:

```
df1=df[df["New Cluster"]==0]
df2=df[df["New Cluster"]==1]
df3=df[df["New Cluster"]==2]
plt.scatter(df1["Quantity"],df1["UnitPrice"],color="red")
plt.scatter(df2["Quantity"],df2["UnitPrice"],color="green")
plt.scatter(df3["Quantity"],df3["UnitPrice"],color="blue")
plt.xlabel("Age")
plt.ylabel("Income($)")
```

Out[32]:

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



In [33]:
```
km.cluster_centers_
```

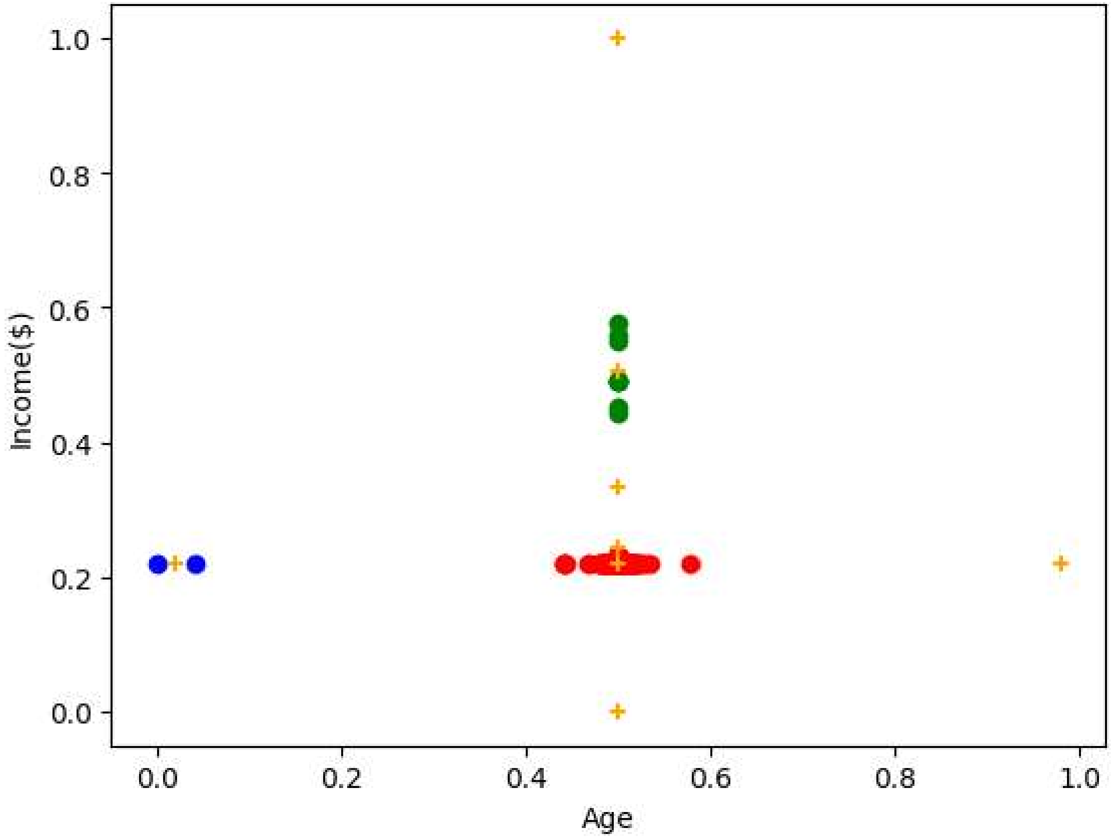
Out[33]:
```
array([[0.50005899, 0.22117195],
       [0.49999657, 0.50519622],
       [0.02092722, 0.22113061],
       [0.49999588, 0.33389462],
       [0.97907278, 0.22113061],
       [0.49999383, 1.        ],
       [0.50000617, 0.        ],
       [0.50000432, 0.24394336]])
```

In [34]:

```
df1=df[df["New Cluster"]==0]
df2=df[df["New Cluster"]==1]
df3=df[df["New Cluster"]==2]
plt.scatter(df1["Quantity"],df1["UnitPrice"],color="red")
plt.scatter(df2["Quantity"],df2["UnitPrice"],color="green")
plt.scatter(df3["Quantity"],df3["UnitPrice"],color="blue")
plt.scatter(km.cluster_centers_[:,0],km.cluster_centers_
            [:,1],color="orange",marker="+")
plt.xlabel("Age")
plt.ylabel("Income($)")
```

Out[34]:

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



In [35]:

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

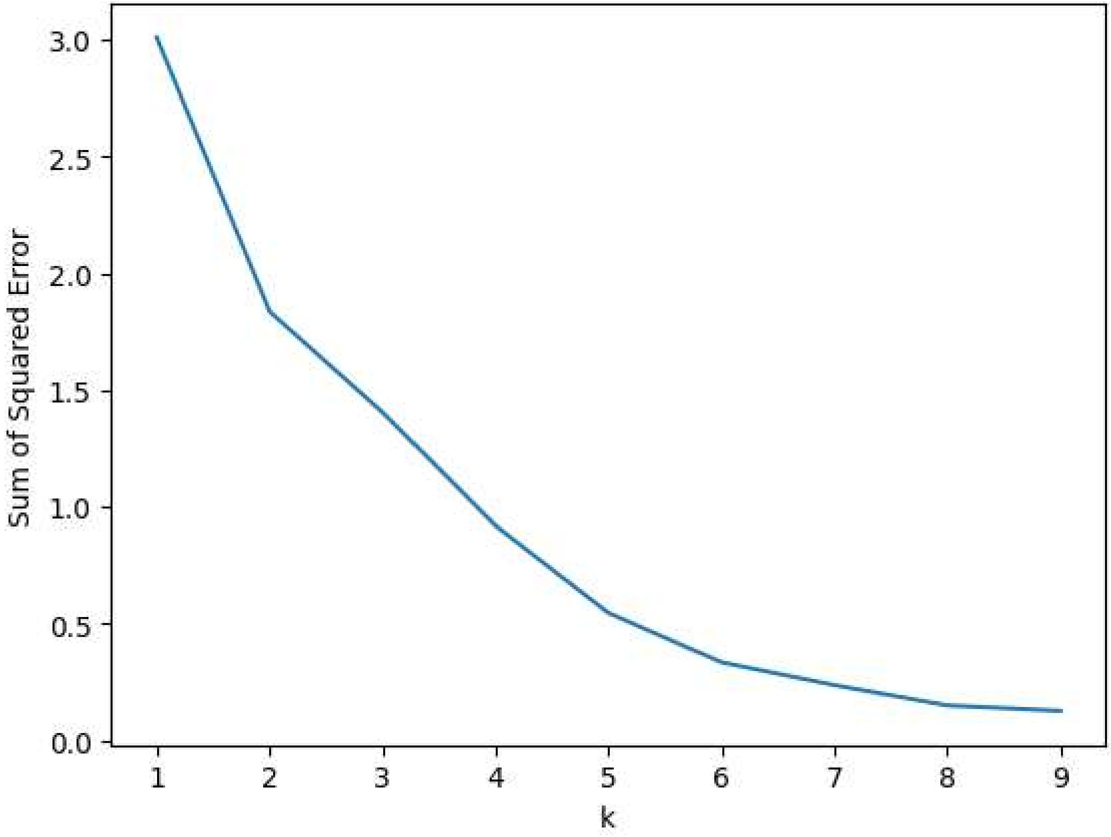
```
C:\Users\my pc\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\cluster\_kmeans.py:870: Fut
ureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` ex
plicitly to suppress the warning
  warnings.warn(
C:\Users\my pc\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\cluster\_kmeans.py:870: Fut
ureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` ex
plicitly to suppress the warning
  warnings.warn(
C:\Users\my pc\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\cluster\_kmeans.py:870: Fut
ureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` ex
plicitly to suppress the warning
  warnings.warn(
C:\Users\my pc\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\cluster\_kmeans.py:870: Fut
ureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` ex
plicitly to suppress the warning
  warnings.warn(
C:\Users\my pc\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\cluster\_kmeans.py:870: Fut
ureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` ex
plicitly to suppress the warning
  warnings.warn(
C:\Users\my pc\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\cluster\_kmeans.py:870: Fut
ureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` ex
plicitly to suppress the warning
  warnings.warn(
C:\Users\my pc\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\cluster\_kmeans.py:870: Fut
ureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` ex
plicitly to suppress the warning
  warnings.warn(
C:\Users\my pc\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\cluster\_kmeans.py:870: Fut
ureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` ex
plicitly to suppress the warning
  warnings.warn(
C:\Users\my pc\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\cluster\_kmeans.py:870: Fut
ureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` ex
plicitly to suppress the warning
  warnings.warn(
```

In [36]:

```
plt.plot(k_rng,sse)
plt.xlabel("k")
plt.ylabel("Sum of Squared Error")
```

Out[36]:

```
Text(0, 0.5, 'Sum of Squared Error')
```



Conclusion:

Here we have clustered for the columns "Quantity","UnitPrice".We have founded clusters for two times, the new clusters formed are 0.Even though ,they are goodly clustered without any error