PROJECT 3

Problem Statement: Which Model is suitable for 100Years_RainfallDataset

Importing Libraries

In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

Reading Data

In [2]:
```
alldf=pd.read_csv(r"C:\Users\P. VIJAY KUMAR\Downloads\100Years_RainfallDat
alldf
```

Out[2]:

	SUBDIVISION	YEAR	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT
0	ANDAMAN & NICOBAR ISLANDS	1901	49.2	87.1	29.2	2.3	528.8	517.5	365.1	481.1	332.6	388.5
1	ANDAMAN & NICOBAR ISLANDS	1902	0.0	159.8	12.2	0.0	446.1	537.1	228.9	753.7	666.2	197.2
2	ANDAMAN & NICOBAR ISLANDS	1903	12.7	144.0	0.0	1.0	235.1	479.9	728.4	326.7	339.0	181.2
3	ANDAMAN & NICOBAR ISLANDS	1904	9.4	14.7	0.0	202.4	304.5	495.1	502.0	160.1	820.4	222.2
4	ANDAMAN & NICOBAR ISLANDS	1905	1.3	0.0	3.3	26.9	279.5	628.7	368.7	330.5	297.0	260.7
...	...	...	...	...	...	...	...	...	...	...	...	...
4111	LAKSHADWEEP	2011	5.1	2.8	3.1	85.9	107.2	153.6	350.2	254.0	255.2	117.4
4112	LAKSHADWEEP	2012	19.2	0.1	1.6	76.8	21.2	327.0	231.5	381.2	179.8	145.9
4113	LAKSHADWEEP	2013	26.2	34.4	37.5	5.3	88.3	426.2	296.4	154.4	180.0	72.8
4114	LAKSHADWEEP	2014	53.2	16.1	4.4	14.9	57.4	244.1	116.1	466.1	132.2	169.2
4115	LAKSHADWEEP	2015	2.2	0.5	3.7	87.1	133.1	296.6	257.5	146.4	160.4	165.4

4116 rows × 19 columns

Data Preprocessing

In [3]:
```
1 alldf.head()
```

Out[3]:

	SUBDIVISION	YEAR	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV
0	ANDAMAN & NICOBAR ISLANDS	1901	49.2	87.1	29.2	2.3	528.8	517.5	365.1	481.1	332.6	388.5	558.2
1	ANDAMAN & NICOBAR ISLANDS	1902	0.0	159.8	12.2	0.0	446.1	537.1	228.9	753.7	666.2	197.2	359.0
2	ANDAMAN & NICOBAR ISLANDS	1903	12.7	144.0	0.0	1.0	235.1	479.9	728.4	326.7	339.0	181.2	284.4
3	ANDAMAN & NICOBAR ISLANDS	1904	9.4	14.7	0.0	202.4	304.5	495.1	502.0	160.1	820.4	222.2	308.7
4	ANDAMAN & NICOBAR ISLANDS	1905	1.3	0.0	3.3	26.9	279.5	628.7	368.7	330.5	297.0	260.7	25.4

In [4]:
```
1 alldf.tail()
```

Out[4]:

	SUBDIVISION	YEAR	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	
4111	LAKSHADWEEP	2011	5.1	2.8	3.1	85.9	107.2	153.6	350.2	254.0	255.2	117.4	1
4112	LAKSHADWEEP	2012	19.2	0.1	1.6	76.8	21.2	327.0	231.5	381.2	179.8	145.9	
4113	LAKSHADWEEP	2013	26.2	34.4	37.5	5.3	88.3	426.2	296.4	154.4	180.0	72.8	
4114	LAKSHADWEEP	2014	53.2	16.1	4.4	14.9	57.4	244.1	116.1	466.1	132.2	169.2	
4115	LAKSHADWEEP	2015	2.2	0.5	3.7	87.1	133.1	296.6	257.5	146.4	160.4	165.4	2

In [5]:
```
1 alldf.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4116 entries, 0 to 4115
Data columns (total 19 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   SUBDIVISION  4116 non-null   object
 1   YEAR         4116 non-null   int64
 2   JAN          4112 non-null   float64
 3   FEB          4113 non-null   float64
 4   MAR          4110 non-null   float64
 5   APR          4112 non-null   float64
 6   MAY          4113 non-null   float64
 7   JUN          4111 non-null   float64
 8   JUL          4109 non-null   float64
 9   AUG          4112 non-null   float64
 10  SEP          4110 non-null   float64
 11  OCT          4109 non-null   float64
 12  NOV          4105 non-null   float64
 13  DEC          4106 non-null   float64
 14  ANNUAL       4090 non-null   float64
 15  Jan-Feb      4110 non-null   float64
 16  Mar-May      4107 non-null   float64
 17  Jun-Sep      4106 non-null   float64
 18  Oct-Dec      4103 non-null   float64
dtypes: float64(17), int64(1), object(1)
memory usage: 611.1+ KB
```

In [6]:
```
1 alldf.shape
```

Out[6]:
```
(4116, 19)
```

In [7]:
```
1 alldf.isnull().sum()
```

Out[7]:
```
SUBDIVISION     0
YEAR            0
JAN             4
FEB             3
MAR             6
APR             4
MAY             3
JUN             5
JUL             7
AUG             4
SEP             6
OCT             7
NOV            11
DEC            10
ANNUAL         26
Jan-Feb         6
Mar-May         9
Jun-Sep        10
Oct-Dec        13
dtype: int64
```

In [8]:
```
1 alldf.fillna(method='ffill',inplace=True)
```

In [9]:
```
1 alldf.isnull().sum()
```

Out[9]:
```
SUBDIVISION    0
YEAR           0
JAN            0
FEB            0
MAR            0
APR            0
MAY            0
JUN            0
JUL            0
AUG            0
SEP            0
OCT            0
NOV            0
DEC            0
ANNUAL         0
Jan-Feb        0
Mar-May        0
Jun-Sep        0
Oct-Dec        0
dtype: int64
```

In [10]:
```
columns_to_drop=alldf[['Jan-Feb', 'Mar-May','Jun-Sep', 'Oct-Dec']]
alldf=alldf.drop(columns_to_drop,axis=1)
alldf
```

Out[10]:

	SUBDIVISION	YEAR	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT
0	ANDAMAN & NICOBAR ISLANDS	1901	49.2	87.1	29.2	2.3	528.8	517.5	365.1	481.1	332.6	388.5
1	ANDAMAN & NICOBAR ISLANDS	1902	0.0	159.8	12.2	0.0	446.1	537.1	228.9	753.7	666.2	197.2
2	ANDAMAN & NICOBAR ISLANDS	1903	12.7	144.0	0.0	1.0	235.1	479.9	728.4	326.7	339.0	181.2
3	ANDAMAN & NICOBAR ISLANDS	1904	9.4	14.7	0.0	202.4	304.5	495.1	502.0	160.1	820.4	222.2
4	ANDAMAN & NICOBAR ISLANDS	1905	1.3	0.0	3.3	26.9	279.5	628.7	368.7	330.5	297.0	260.7
...	...	...	...	...	...	...	...	...	...	...	...	...
4111	LAKSHADWEEP	2011	5.1	2.8	3.1	85.9	107.2	153.6	350.2	254.0	255.2	117.4
4112	LAKSHADWEEP	2012	19.2	0.1	1.6	76.8	21.2	327.0	231.5	381.2	179.8	145.9
4113	LAKSHADWEEP	2013	26.2	34.4	37.5	5.3	88.3	426.2	296.4	154.4	180.0	72.8
4114	LAKSHADWEEP	2014	53.2	16.1	4.4	14.9	57.4	244.1	116.1	466.1	132.2	169.2
4115	LAKSHADWEEP	2015	2.2	0.5	3.7	87.1	133.1	296.6	257.5	146.4	160.4	165.4

4116 rows × 15 columns

In [11]:
```
1 alldf["SUBDIVISION"].value_counts()
```

Out[11]:
```
SUBDIVISION
WEST MADHYA PRADESH                    115
EAST RAJASTHAN                         115
COASTAL KARNATAKA                      115
TAMIL NADU                             115
RAYALSEEMA                             115
TELANGANA                              115
COASTAL ANDHRA PRADESH                 115
CHHATTISGARH                           115
VIDARBHA                               115
MATATHWADA                             115
MADHYA MAHARASHTRA                     115
KONKAN & GOA                           115
SAURASHTRA & KUTCH                     115
GUJARAT REGION                         115
EAST MADHYA PRADESH                    115
KERALA                                 115
WEST RAJASTHAN                         115
SOUTH INTERIOR KARNATAKA               115
JAMMU & KASHMIR                        115
HIMACHAL PRADESH                       115
PUNJAB                                 115
HARYANA DELHI & CHANDIGARH             115
UTTARAKHAND                            115
WEST UTTAR PRADESH                     115
EAST UTTAR PRADESH                     115
BIHAR                                  115
JHARKHAND                              115
ORISSA                                 115
GANGETIC WEST BENGAL                   115
SUB HIMALAYAN WEST BENGAL & SIKKIM     115
NAGA MANI MIZO TRIPURA                 115
ASSAM & MEGHALAYA                      115
NORTH INTERIOR KARNATAKA               115
LAKSHADWEEP                            114
ANDAMAN & NICOBAR ISLANDS              110
ARUNACHAL PRADESH                       97
Name: count, dtype: int64
```

In [12]:

```
states={"SUBDIVISION":{
"WEST MADHYA PRADESH":1,
"EAST RAJASTHAN":2,
"COASTAL KARNATAKA":3,
"TAMIL NADU":4,
"RAYALSEEMA":5,
"TELANGANA":6,
"COASTAL ANDHRA PRADESH":7,
"CHHATTISGARH":8,
"VIDARBHA":9,
"MATATHWADA":10,
"MADHYA MAHARASHTRA":11,
"KONKAN & GOA":12,
"SAURASHTRA & KUTCH":13,
"GUJARAT REGION":14,
"EAST MADHYA PRADESH":15,
"KERALA":16,
"WEST RAJASTHAN":17,
"SOUTH INTERIOR KARNATAKA":18,
"JAMMU & KASHMIR":19,
"HIMACHAL PRADESH":20,
"PUNJAB":21,
"HARYANA DELHI & CHANDIGARH":22,
"UTTARAKHAND":23,
"WEST UTTAR PRADESH":24,
"EAST UTTAR PRADESH":25,
"BIHAR":26,
"JHARKHAND":27,
"ORISSA":28,
"GANGETIC WEST BENGAL":29,
"SUB HIMALAYAN WEST BENGAL & SIKKIM":30,
"NAGA MANI MIZO TRIPURA":31,
"ASSAM & MEGHALAYA":32,
"NORTH INTERIOR KARNATAKA":33,
"LAKSHADWEEP":34,
"ANDAMAN & NICOBAR ISLANDS":35,
"ARUNACHAL PRADESH":36}}
alldf=alldf.replace(states)
alldf
```

Out[12]:

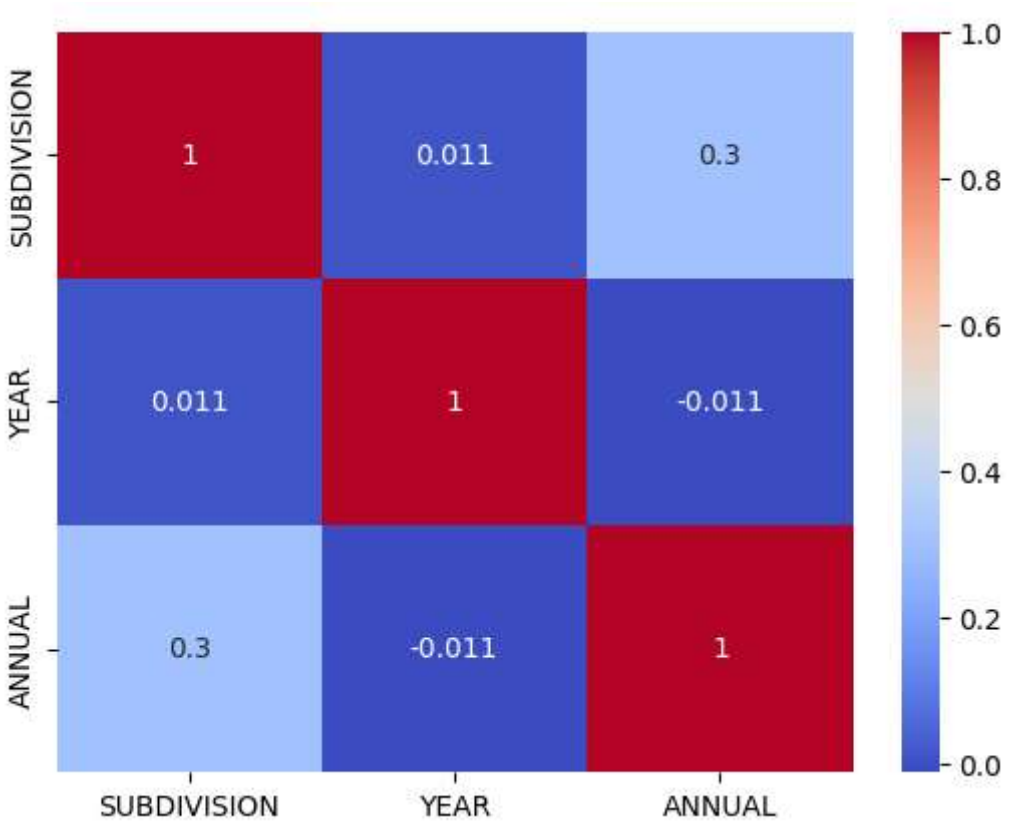
	SUBDIVISION	YEAR	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	N
0	35	1901	49.2	87.1	29.2	2.3	528.8	517.5	365.1	481.1	332.6	388.5	5
1	35	1902	0.0	159.8	12.2	0.0	446.1	537.1	228.9	753.7	666.2	197.2	3
2	35	1903	12.7	144.0	0.0	1.0	235.1	479.9	728.4	326.7	339.0	181.2	2
3	35	1904	9.4	14.7	0.0	202.4	304.5	495.1	502.0	160.1	820.4	222.2	3
4	35	1905	1.3	0.0	3.3	26.9	279.5	628.7	368.7	330.5	297.0	260.7	
...	...	...	...	...	...	...	...	...	...	...	...	...	
4111	34	2011	5.1	2.8	3.1	85.9	107.2	153.6	350.2	254.0	255.2	117.4	1
4112	34	2012	19.2	0.1	1.6	76.8	21.2	327.0	231.5	381.2	179.8	145.9	
4113	34	2013	26.2	34.4	37.5	5.3	88.3	426.2	296.4	154.4	180.0	72.8	
4114	34	2014	53.2	16.1	4.4	14.9	57.4	244.1	116.1	466.1	132.2	169.2	
4115	34	2015	2.2	0.5	3.7	87.1	133.1	296.6	257.5	146.4	160.4	165.4	2

4116 rows × 15 columns

In [14]:

```
1 columns=alldf[["SUBDIVISION","YEAR","ANNUAL"]]
2 subset=columns.corr()
3 sns.heatmap(subset,annot=True,cmap='coolwarm')
```

Out[14]: <Axes: >



Feature Scaling: Splitting Dataset into training and testing dataset

In [24]:
```
x=np.array(alldf["SUBDIVISION"]).reshape(-1,1)
y=np.array(alldf["ANNUAL"]).reshape(-1,1)
```

In [25]:
```
from sklearn.model_selection import train_test_split

```

In [26]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_s
```

Applying Linear Regression

In [27]:
```
from sklearn.linear_model import LinearRegression
```

In [28]:
```
lr=LinearRegression()
```

In [29]:
```
lr.fit(x_train,y_train)
```

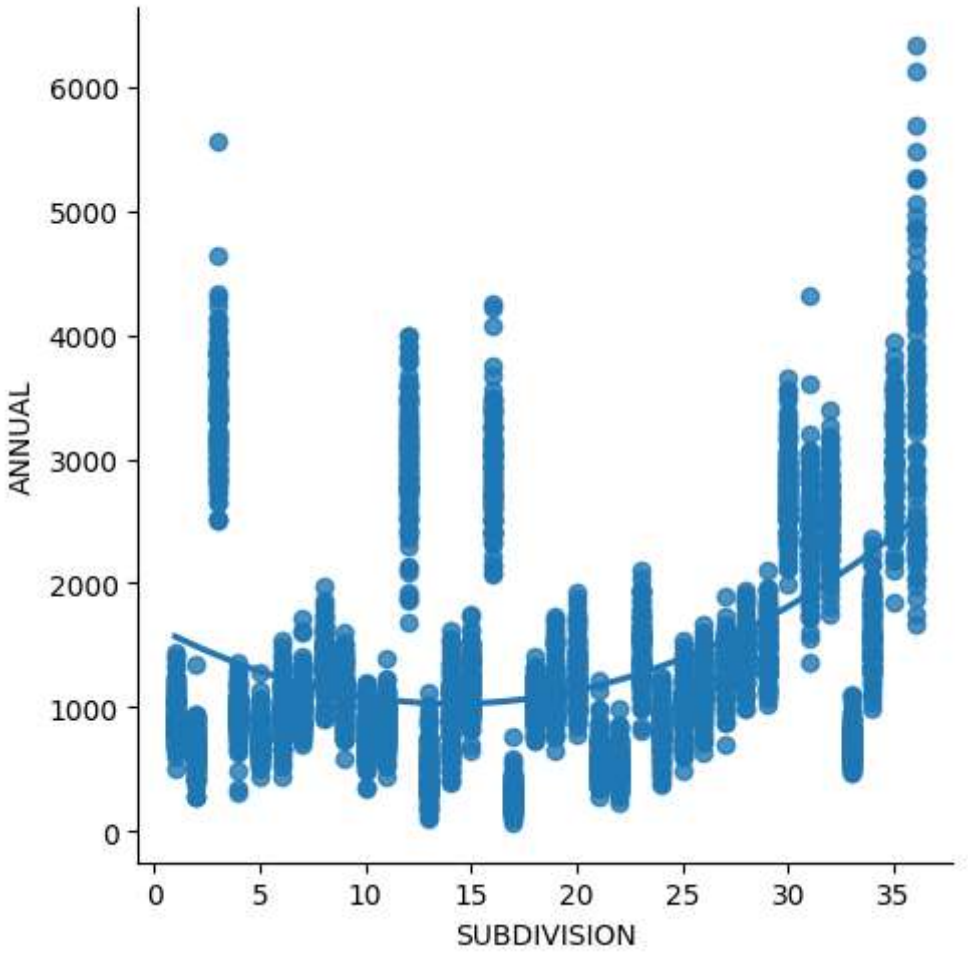
Out[29]:
```
LinearRegression()
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [30]:
```
lr.score(x_test,y_test)
```

Out[30]:
```
0.087360906001137
```

In [32]:
```
1 sns.lmplot(x="SUBDIVISION",y="ANNUAL",data=alldf,order=2,ci=None)
```

Out[32]: <seaborn.axisgrid.FacetGrid at 0x197a8937130>



Applying Logistic Regression

In [33]:
```
1 from sklearn.linear_model import LogisticRegression
```

In [34]:
```
1 lg=LogisticRegression()
```

In [36]:
```
1 X=y
2 Y=x
```

In [37]:
```
1 x_train,x_test,y_train,y_test=train_test_split(X,Y,test_size=0.3,random_s
```

In [38]:
```
1 lg.fit(x_train,y_train)
```

```
C:\Users\P. VIJAY KUMAR\AppData\Roaming\Python\Python310\site-packages\sklear
n\utils\validation.py:1143: DataConversionWarning: A column-vector y was pass
ed when a 1d array was expected. Please change the shape of y to (n_samples,
), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\P. VIJAY KUMAR\AppData\Roaming\Python\Python310\site-packages\sklear
n\linear_model\_logistic.py:458: ConvergenceWarning: lbfgs failed to converge
(status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://sciki
t-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regres
sion (https://scikit-learn.org/stable/modules/linear_model.html#logistic-regr
ession)
  n_iter_i = _check_optimize_result(
```

Out[38]:
```
LogisticRegression()
```

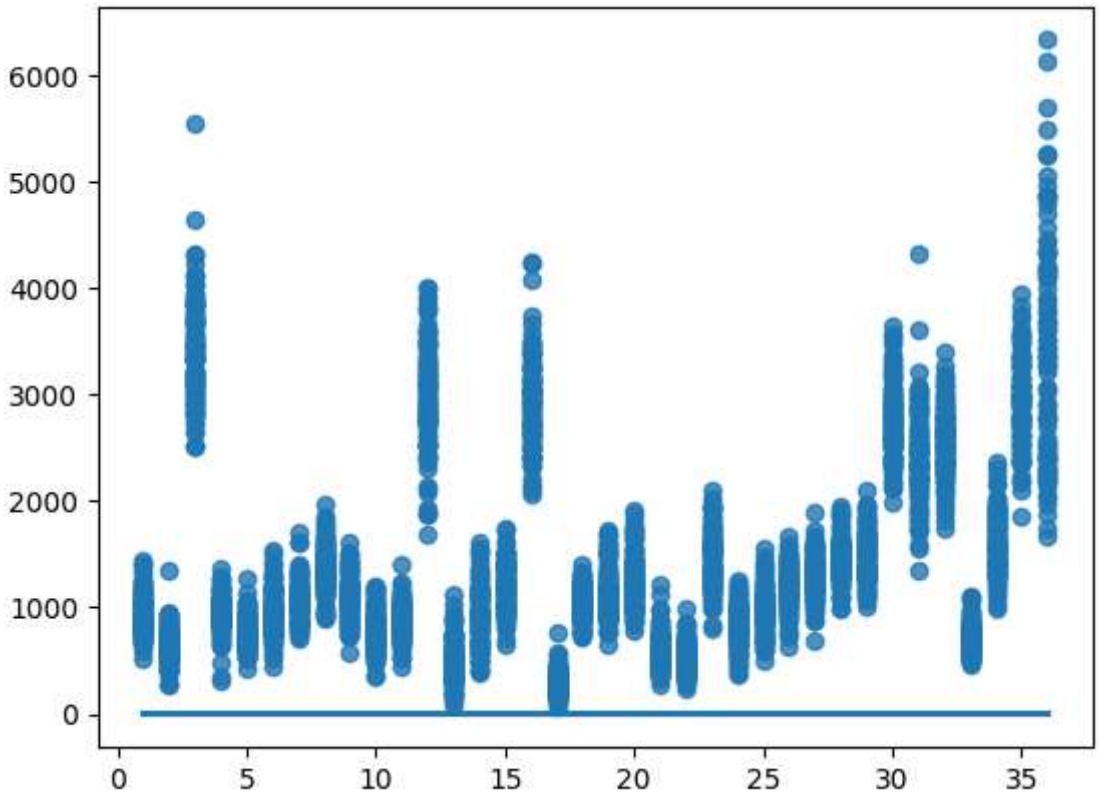
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [39]:
```
1 lg.score(x_test,y_test)
```

Out[39]:
```
0.1408906882591093
```

In [41]:
```
1 sns.regplot(x=x,y=y,data=alldf,logistic=True,ci=None)
```

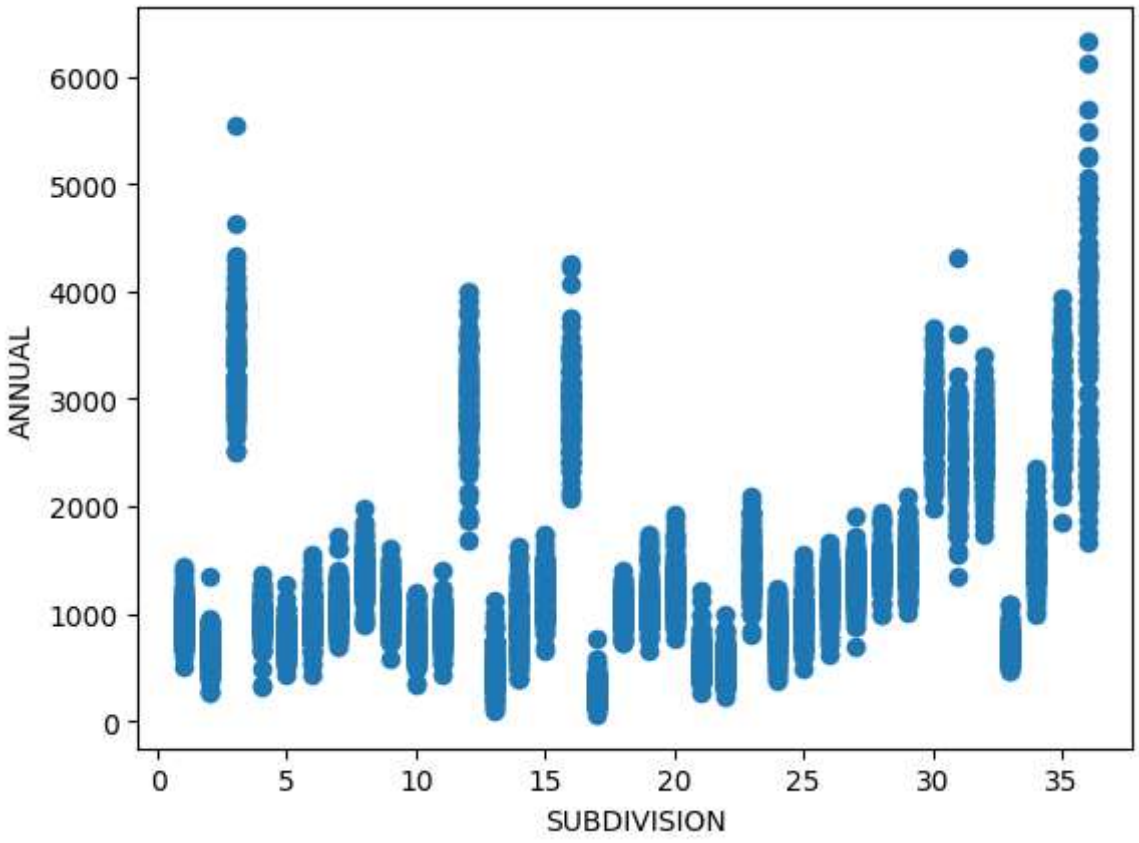
Out[41]: <Axes: >



Applying KMeans

In [42]:
```
df=alldf
plt.scatter(df["SUBDIVISION"],df["ANNUAL"])
plt.xlabel("SUBDIVISION")
plt.ylabel("ANNUAL")
```

Out[42]:
```
Text(0, 0.5, 'ANNUAL')
```



In [43]:
```
from sklearn.cluster import KMeans
km=KMeans()
km
```

Out[43]:
```
KMeans()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [44]:
```
y_predicted=km.fit_predict(df[["SUBDIVISION","ANNUAL"]])
y_predicted
```

```
C:\Users\P. VIJAY KUMAR\AppData\Roaming\Python\Python310\site-packages\sklear
n\cluster\_kmeans.py:870: FutureWarning: The default value of `n_init` will c
hange from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppr
ess the warning
  warnings.warn(
```

Out[44]:
```
array([6, 6, 1, ..., 4, 4, 4])
```

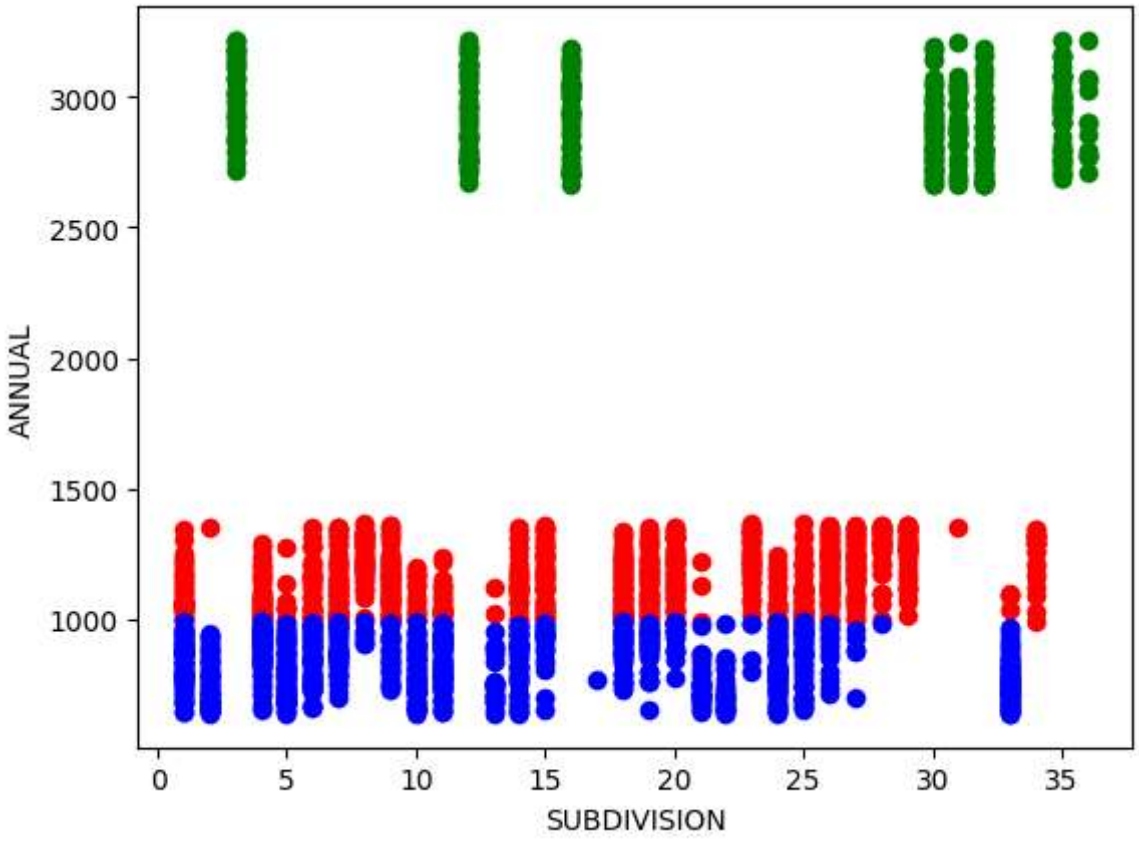
In [45]:
```
df["cluster"]=y_predicted
df.head()
```

Out[45]:

	SUBDIVISION	YEAR	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV
0	35	1901	49.2	87.1	29.2	2.3	528.8	517.5	365.1	481.1	332.6	388.5	558.2
1	35	1902	0.0	159.8	12.2	0.0	446.1	537.1	228.9	753.7	666.2	197.2	359.0
2	35	1903	12.7	144.0	0.0	1.0	235.1	479.9	728.4	326.7	339.0	181.2	284.4
3	35	1904	9.4	14.7	0.0	202.4	304.5	495.1	502.0	160.1	820.4	222.2	308.7
4	35	1905	1.3	0.0	3.3	26.9	279.5	628.7	368.7	330.5	297.0	260.7	25.4

In [46]:
```
df1=df[df.cluster==0]
df2=df[df.cluster==1]
df3=df[df.cluster==2]
plt.scatter(df1["SUBDIVISION"],df1["ANNUAL"],color="red")
plt.scatter(df2["SUBDIVISION"],df2["ANNUAL"],color="green")
plt.scatter(df3["SUBDIVISION"],df3["ANNUAL"],color="blue")
plt.xlabel("SUBDIVISION")
plt.ylabel("ANNUAL")
```

Out[46]: Text(0, 0.5, 'ANNUAL')



In [47]:
```
from sklearn.preprocessing import MinMaxScaler
scaler=MinMaxScaler()
scaler.fit(df[["ANNUAL"]])
df["ANNUAL"]=scaler.transform(df[["ANNUAL"]])
df.head()
```

Out[47]:

	SUBDIVISION	YEAR	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV
0	35	1901	49.2	87.1	29.2	2.3	528.8	517.5	365.1	481.1	332.6	388.5	558.2
1	35	1902	0.0	159.8	12.2	0.0	446.1	537.1	228.9	753.7	666.2	197.2	359.0
2	35	1903	12.7	144.0	0.0	1.0	235.1	479.9	728.4	326.7	339.0	181.2	284.4
3	35	1904	9.4	14.7	0.0	202.4	304.5	495.1	502.0	160.1	820.4	222.2	308.7
4	35	1905	1.3	0.0	3.3	26.9	279.5	628.7	368.7	330.5	297.0	260.7	25.4

In [48]:
```
scaler.fit(df[["SUBDIVISION"]])
df["SUBDIVISION"]=scaler.transform(df[["SUBDIVISION"]])
df.head()
```

Out[48]:

	SUBDIVISION	YEAR	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV
0	0.971429	1901	49.2	87.1	29.2	2.3	528.8	517.5	365.1	481.1	332.6	388.5	558.2
1	0.971429	1902	0.0	159.8	12.2	0.0	446.1	537.1	228.9	753.7	666.2	197.2	359.0
2	0.971429	1903	12.7	144.0	0.0	1.0	235.1	479.9	728.4	326.7	339.0	181.2	284.4
3	0.971429	1904	9.4	14.7	0.0	202.4	304.5	495.1	502.0	160.1	820.4	222.2	308.7
4	0.971429	1905	1.3	0.0	3.3	26.9	279.5	628.7	368.7	330.5	297.0	260.7	25.4

In [49]:
```
y_predicted=km.fit_predict(df[["SUBDIVISION","ANNUAL"]])
y_predicted
```

```
C:\Users\P. VIJAY KUMAR\AppData\Roaming\Python\Python310\site-packages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
```

Out[49]:
```
array([6, 6, 6, ..., 0, 0, 0])
```

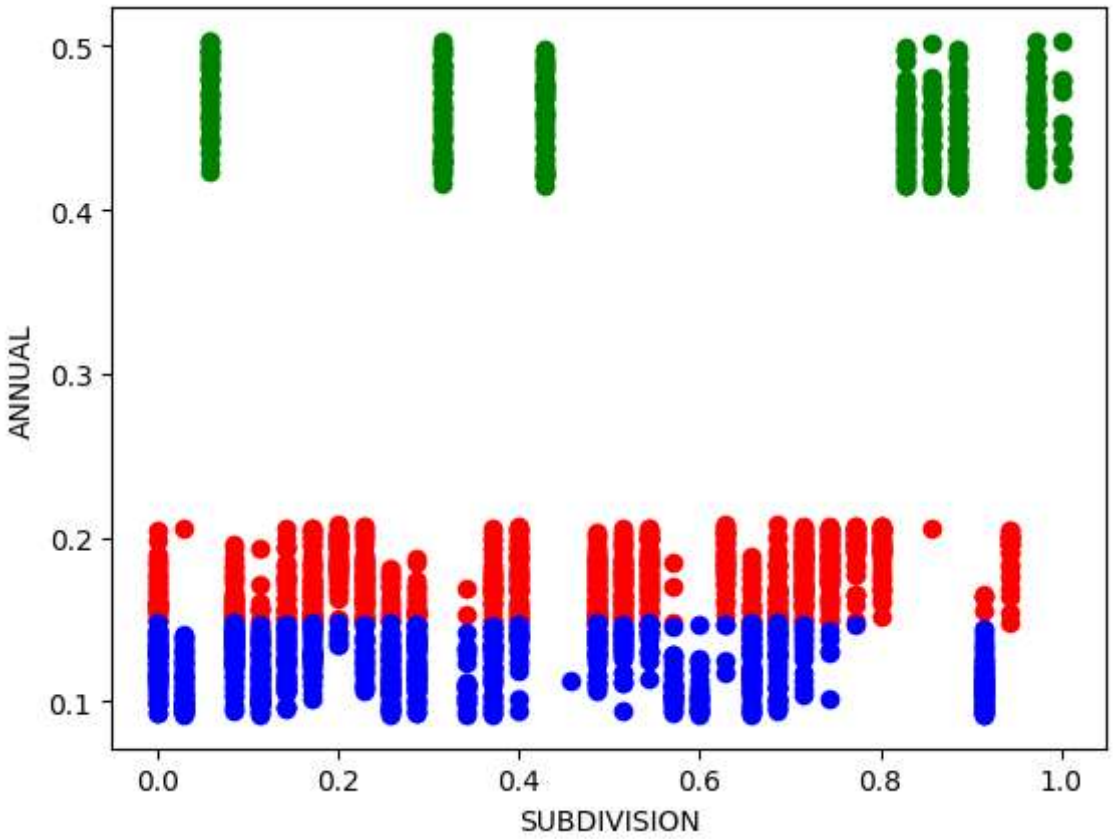
In [50]:
```
df["New Cluster"]=y_predicted
df.head()
```

Out[50]:

	SUBDIVISION	YEAR	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV
0	0.971429	1901	49.2	87.1	29.2	2.3	528.8	517.5	365.1	481.1	332.6	388.5	558.2
1	0.971429	1902	0.0	159.8	12.2	0.0	446.1	537.1	228.9	753.7	666.2	197.2	359.0
2	0.971429	1903	12.7	144.0	0.0	1.0	235.1	479.9	728.4	326.7	339.0	181.2	284.4
3	0.971429	1904	9.4	14.7	0.0	202.4	304.5	495.1	502.0	160.1	820.4	222.2	308.7
4	0.971429	1905	1.3	0.0	3.3	26.9	279.5	628.7	368.7	330.5	297.0	260.7	25.4

In [51]:
```
df1=df[df.cluster==0]
df2=df[df.cluster==1]
df3=df[df.cluster==2]
plt.scatter(df1["SUBDIVISION"],df1["ANNUAL"],color="red")
plt.scatter(df2["SUBDIVISION"],df2["ANNUAL"],color="green")
plt.scatter(df3["SUBDIVISION"],df3["ANNUAL"],color="blue")
plt.xlabel("SUBDIVISION")
plt.ylabel("ANNUAL")
```

Out[51]: Text(0, 0.5, 'ANNUAL')

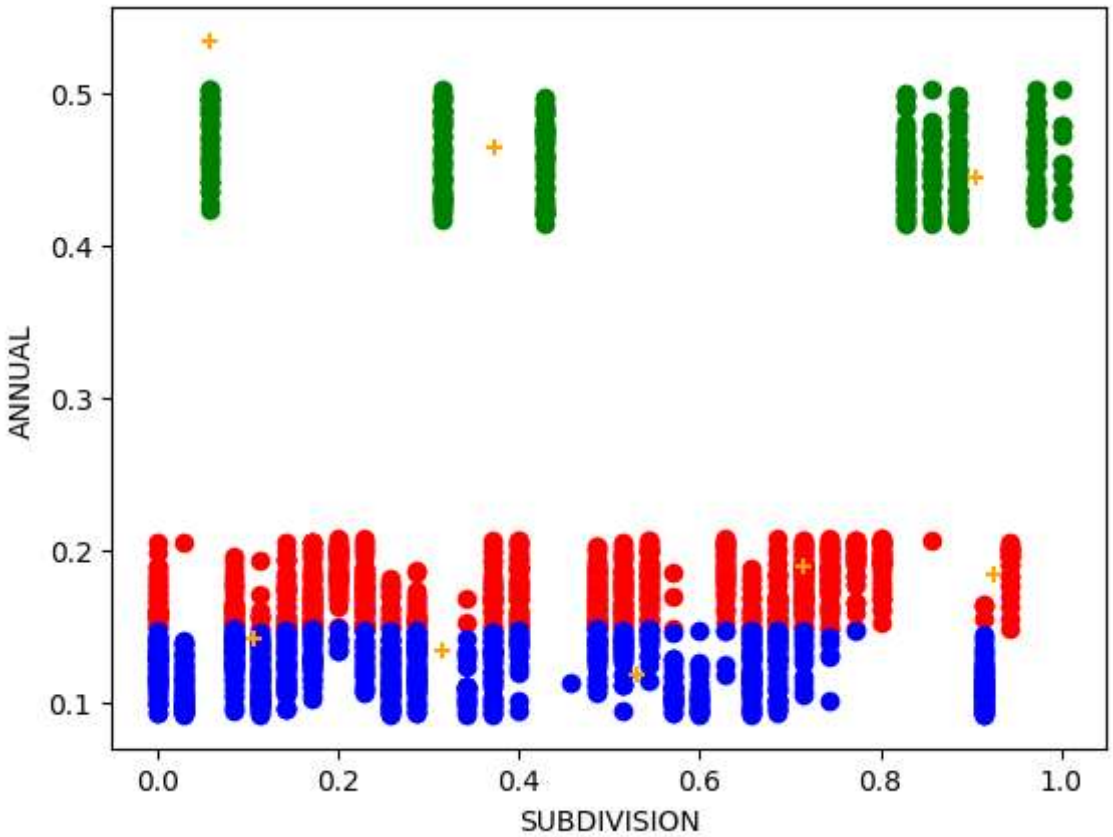


In [52]:
```
1 km.cluster_centers_
```

Out[52]:
```
array([[0.92433653, 0.18350317],
       [0.10612245, 0.1424822 ],
       [0.5295977 , 0.1188574 ],
       [0.05714286, 0.53446579],
       [0.37269841, 0.46481389],
       [0.71467956, 0.18961902],
       [0.90575209, 0.44434818],
       [0.31428571, 0.13431543]])
```

In [54]:
```
1 df1=df[df.cluster==0]
2 df2=df[df.cluster==1]
3 df3=df[df.cluster==2]
4 plt.scatter(df1["SUBDIVISION"],df1["ANNUAL"],color="red")
5 plt.scatter(df2["SUBDIVISION"],df2["ANNUAL"],color="green")
6 plt.scatter(df3["SUBDIVISION"],df3["ANNUAL"],color="blue")
7 plt.scatter(km.cluster_centers_[:,0],km.cluster_centers_[:,1],color="oran
8 plt.xlabel("SUBDIVISION")
9 plt.ylabel("ANNUAL")
```

Out[54]:
```
Text(0, 0.5, 'ANNUAL')
```



In [55]:
```
1 k_rng=range(1,10)
2 sse=[]
```

In [56]:

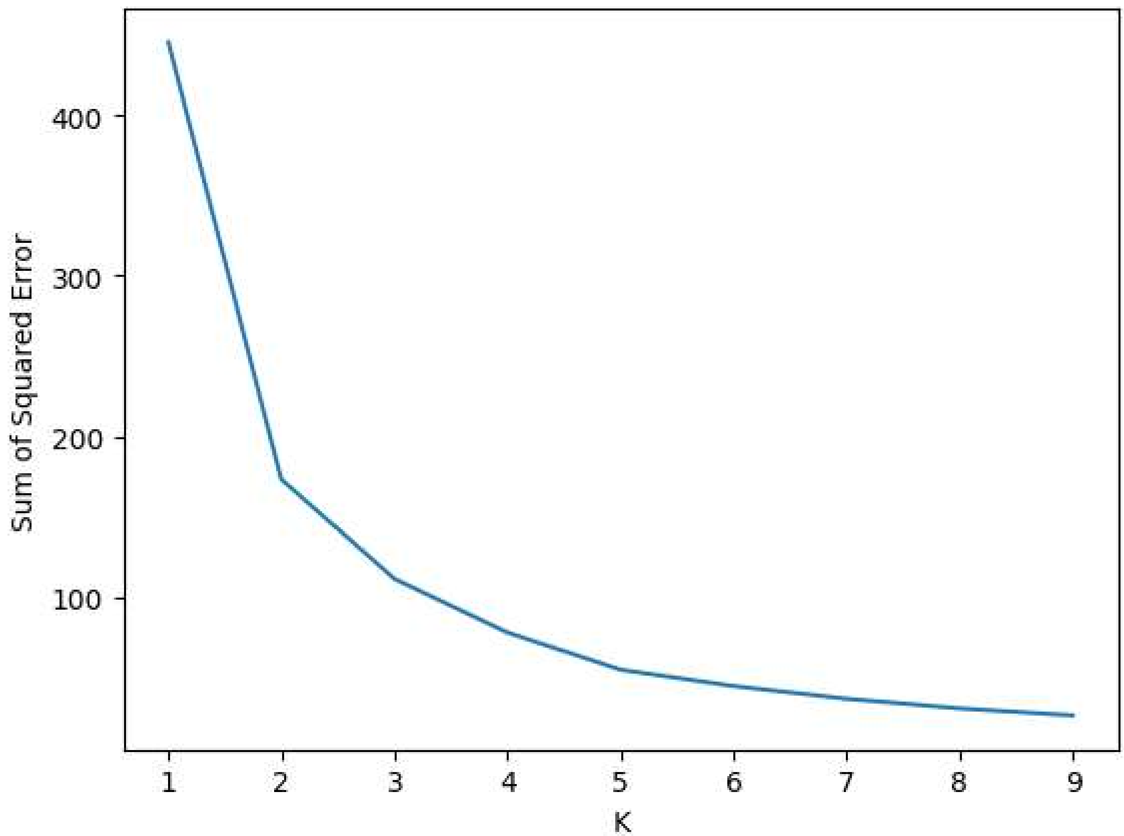
```
for k in k_rng:
 km=KMeans(n_clusters=k)
 km.fit(df[["SUBDIVISION","ANNUAL"]])
 sse.append(km.inertia_)
#km.inertia_ will give you the value of sum of square error
print(sse)
plt.plot(k_rng,sse)
plt.xlabel("K")
plt.ylabel("Sum of Squared Error")
```

```
C:\Users\P. VIJAY KUMAR\AppData\Roaming\Python\Python310\site-packages\sklear
n\cluster\_kmeans.py:870: FutureWarning: The default value of `n_init` will c
hange from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppr
ess the warning
  warnings.warn(
C:\Users\P. VIJAY KUMAR\AppData\Roaming\Python\Python310\site-packages\sklear
n\cluster\_kmeans.py:870: FutureWarning: The default value of `n_init` will c
hange from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppr
ess the warning
  warnings.warn(
C:\Users\P. VIJAY KUMAR\AppData\Roaming\Python\Python310\site-packages\sklear
n\cluster\_kmeans.py:870: FutureWarning: The default value of `n_init` will c
hange from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppr
ess the warning
  warnings.warn(
C:\Users\P. VIJAY KUMAR\AppData\Roaming\Python\Python310\site-packages\sklear
n\cluster\_kmeans.py:870: FutureWarning: The default value of `n_init` will c
hange from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppr
ess the warning
  warnings.warn(
C:\Users\P. VIJAY KUMAR\AppData\Roaming\Python\Python310\site-packages\sklear
n\cluster\_kmeans.py:870: FutureWarning: The default value of `n_init` will c
hange from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppr
ess the warning
  warnings.warn(
C:\Users\P. VIJAY KUMAR\AppData\Roaming\Python\Python310\site-packages\sklear
n\cluster\_kmeans.py:870: FutureWarning: The default value of `n_init` will c
hange from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppr
ess the warning
  warnings.warn(

[445.1192317411816, 173.33047587054514, 111.4773199850639, 78.08560964158761,
54.82822922052405, 44.672277357906495, 36.731015403194476, 30.74287431350261,
26.363693798039385]
```

```
C:\Users\P. VIJAY KUMAR\AppData\Roaming\Python\Python310\site-packages\sklear
n\cluster\_kmeans.py:870: FutureWarning: The default value of `n_init` will c
hange from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppr
ess the warning
  warnings.warn(
C:\Users\P. VIJAY KUMAR\AppData\Roaming\Python\Python310\site-packages\sklear
n\cluster\_kmeans.py:870: FutureWarning: The default value of `n_init` will c
hange from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppr
ess the warning
  warnings.warn(
C:\Users\P. VIJAY KUMAR\AppData\Roaming\Python\Python310\site-packages\sklear
n\cluster\_kmeans.py:870: FutureWarning: The default value of `n_init` will c
hange from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppr
ess the warning
  warnings.warn(
```

Out[56]: Text(0, 0.5, 'Sum of Squared Error')



CONCLUSION: The KMeans algorithm is best fitted model for given dataset

In [ ]:

```
1
```