In [70]:
```
import pandas as pd
import matplotlib.pyplot as plt
```

In [71]:
```
df=pd.read_csv("D:/Mall_Customers.csv")
```

In [72]:
```
df
```

Out[72]:

	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40
...	...	...	...	...	...
195	196	Female	35	120	79
196	197	Female	45	126	28
197	198	Male	32	126	74
198	199	Male	32	137	18
199	200	Male	30	137	83

200 rows × 5 columns

In [73]:
```
df=df.iloc[:,3:]
```

In [74]:
```
df
```

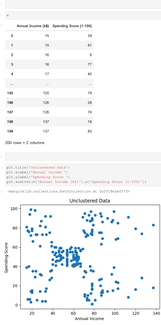
Out[74]:

	Annual Income (k$)	Spending Score (1-100)
0	15	39
1	15	81
2	16	6
3	16	77
4	17	40
...	...	...
195	120	79
196	126	28
197	126	74
198	137	18
199	137	83

200 rows × 2 columns

In [75]:
```
plt.title("Unclustered Data")
plt.xlabel("Annual Income ")
plt.ylabel("Spending Score ")
plt.scatter(df["Annual Income (k$)"],df["Spending Score (1-100)"])
```

Out[75]:
```
<matplotlib.collections.PathCollection at 0x2178c5e0f70>
```



In [76]:
```
from sklearn.cluster import KMeans,AgglomerativeClustering
km=KMeans(n_clusters=5)
```

In [77]:
```
x.shape
```

Out[77]:
```
(200, 2)
```

In [78]:
```
km.fit_predict(x)
```

Out[78]:
```
array([3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3,
       2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3,
       2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 1, 3, 2, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 4, 1, 4, 0, 4, 0, 4, 0, 4,
       0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4,
       0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4,
       0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4,
       0, 4, dtype=int32)
```

In [79]:
```
km.inertia_
```

Out[79]:
```
269346.0739361116
```

In [80]:
```
cost=[]
for i in range(1,101):
    km=KMeans(n_clusters=i)
    km.fit_predict(x)
    cost.append(km.inertia_)
```

In [81]:
```
cost
```

Out[81]:
```
[269981.28,
 181363.59595959596,
 106348.37306211119,
 73679.78903948834,
 44448.45544793369,
 37233.81451071481,
 30241.434615773334,
 25011.839349415256,
 21832.05465925466,
 19664.710988435124,
 17797.923781326944,
 16039.037908201124,
 14848.587713675213,
 13744.267479907672,
 12566.568211368532,
 11664.944581622416,
 10929.948819084216,
 9969.282687953431,
 9323.612903225806,
 8748.528137806075,
 8301.364672364672,
 7881.753968253968,
 7526.966666666666,
 7083.832942973523,
 6688.094276094276,
 6348.673031026252,
 6059.283857168659,
 5804.731762065095,
 5491.719052959501,
 5267.657992565055,
 5035.799837642192,
 4848.132034632035,
 4635.896270396270,
 4467.544444444444,
 4329.930930930931,
 4170.370370370370,
 4037.080952380952,
 3890.465909090909,
 3769.759259259259,
 3641.557539682539,
 3535.832972972973,
 3417.926315789474,
 3320.692017316017,
 3230.334343434343,
 3125.857142857143,
 3037.630769230769,
 2948.777777777777,
 2867.496666666667,
 2789.290719697719,
 2720.333333333333,
 2649.852380952381,
 2583.383838383838,
 2518.666666666667,
 2464.042277922078,
 2404.777777777778,
 2347.896666666667,
 2299.333333333333,

 2251.333333333333,
 2203.333333333333,
 2155.333333333333,
 2107.333333333333,
 2059.333333333333,
 2011.333333333333,
 1963.333333333333,
 1915.333333333333,
 1867.333333333333,
 1819.333333333333,
 1771.333333333333,
 1723.333333333333,
 1675.333333333333,
 1627.333333333333,
 1579.333333333333,
 1531.333333333333,
 1483.333333333333,
 1435.333333333333,
 1387.333333333333,
 1339.333333333333,
 1291.333333333333,
 1243.333333333333,
 1195.333333333333,
 1147.333333333333,
 1099.333333333333,
 1051.333333333333,
 1003.333333333333,
 955.333333333333,
 907.333333333333,
 859.333333333333,
 811.333333333333,
 763.333333333333,
 715.333333333333,
 667.333333333333,
 619.333333333333,
 571.333333333333,
 523.333333333333,
 475.333333333333,
 427.333333333333,
 379.333333333333,
 331.333333333333,
 283.333333333333,
 235.333333333333,
 187.333333333333,
 139.333333333333,
 91.333333333333,
 43.333333333333]
```

In [82]:
```
plt.title("Elbow Method")
plt.xlabel("Value of k")
plt.ylabel("TSSE")
plt.grid()
plt.xticks(range(1,101))
plt.plot(range(1,101),cost,marker=".",color="red")
```

Out[82]:
```
[<matplotlib.lines.Line2D at 0x2c5f4313c20>]
```

In [83]:
```
from sklearn.metrics import silhouette_score
```

In [84]:
```
cost=[]
for i in range(2,101):
    km=KMeans(n_clusters=i)
    labels=km.fit_predict(x)
    cost.append(silhouette_score(x,labels))
    plt.ylabel("Silhouette Score")
```

In [85]:
```
plt.title("Silhouette Method")
plt.xlabel("Value of k")
plt.ylabel("Silhouette Score")
plt.grid()
plt.xticks(range(2,101))
plt.bar(range(2,101),cost,color="red")
```

Out[85]:
```
<BarContainer object of 99 artists>
```

In [86]:
```
km=KMeans(n_clusters=5,random_state=0)
labels=km.fit_predict(x)
```

In [87]:
```
labels
```

Out[87]:
```
array([3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3,
       4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3,
       4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 1, 3, 4, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 2, 0, 2, 0, 2, 0, 2,
       0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2,
       0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2,
       0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2,
       0, 2, dtype=int32)
```

In [88]:
```
plt.figure(figsize=(4,12))
sns=km.cluster_centers_
plt.subplot(3,1,1)
plt.title("Clustered Data")
plt.xlabel("Annual Income ")
plt.ylabel("Spending Score ")
plt.scatter(df["Annual Income (k$)"],df["Spending Score (1-100)"],c=labels)
plt.scatter(sns[:,0],sns[:,1],c=[0,1,2,3,4],s=200,cmap="b")
```

Out[88]:
```
<matplotlib.collections.PathCollection at 0x2c5f4c9e810>
```

In [89]:
```
km.inertia_
```

Out[89]:
```
44448.45544793369
```

In [90]:
```
km.labels_
```

Out[90]:
```
array([3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3,
       4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3,
       4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 1, 3, 4, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 2, 0, 2, 0, 2, 0, 2,
       0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2,
       0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2,
       0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2,
       0, 2, dtype=int32)
```

In [91]:
```
df[labels==0]
```

Out[91]:

	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)
1	2	Male	21	15	81
3	4	Female	23	16	77
5	6	Female	22	17	76
7	8	Female	23	18	94
9	10	Female	30	19	72
11	12	Female	35	19	99
13	14	Female	24	20	77
15	16	Male	22	20	79
17	18	Male	20	21	66
19	20	Female	35	23	98
21	22	Male	25	24	73
23	24	Male	31	25	73
25	26	Male	29	28	82
27	28	Male	35	28	61
29	30	Female	23	29	87
31	32	Female	21	30	73
33	34	Male	18	33	92
35	36	Female	21	33	81
37	38	Female	30	34	73
39	40	Female	20	37	75
41	42	Male	24	38	92
45	46	Female	24	39	65

In [92]:
```
km.predict([[28,87]])
```

```
C:\Users\lif\Applications\Programs\Python\Python310\lib\site-packages\sklearn\base.py:450: UserWarning: X does not have valid feature names, but KMeans was fitted with feature names
  warnings.warn(
```

Out[92]:
```
array([4], dtype=int32)
```

In [93]:

In [98]:
```
EnsureNE(LabelLayer4)
```

Out[98]: