In [74]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
from sklearn.cluster import KMeans
from scipy.spatial.distance import cdist
from scipy.cluster.hierarchy import linkage
import scipy.cluster.hierarchy as sch
from sklearn.cluster import AgglomerativeClustering


import warnings
warnings.filterwarnings('ignore')
```

In [38]:
```
data = pd.read_excel('EastWestAirlines.xlsx',sheet_name='data')
data
```

Out[38]:

	ID#	Balance	Qual_miles	cc1_miles	cc2_miles	cc3_miles	Bonus_miles	Bonus_trans	Fli
0	1	28143	0	1	1	1	174	1	
1	2	19244	0	1	1	1	215	2	
2	3	41354	0	1	1	1	4123	4	
3	4	14776	0	1	1	1	500	1	
4	5	97752	0	4	1	1	43300	26	
...	...	...	...	...	...	...	...	...	
3994	4017	18476	0	1	1	1	8525	4	
3995	4018	64385	0	1	1	1	981	5	
3996	4019	73597	0	3	1	1	25447	8	
3997	4020	54899	0	1	1	1	500	1	
3998	4021	3016	0	1	1	1	0	0	

3999 rows × 12 columns

In [39]:
```
data.shape
```

Out[39]:
```
(3999, 12)
```

In [40]:
```
data.isna().sum()
```

Out[40]:
```
ID#                0
Balance            0
Qual_miles         0
cc1_miles          0
cc2_miles          0
cc3_miles          0
Bonus_miles        0
Bonus_trans        0
Flight_miles_12mo  0
Flight_trans_12    0
Days_since_enroll  0
Award?             0
dtype: int64
```

In [41]:
```
data.dtypes
```

Out[41]:
```
ID#                int64
Balance            int64
Qual_miles         int64
cc1_miles          int64
cc2_miles          int64
cc3_miles          int64
Bonus_miles        int64
Bonus_trans        int64
Flight_miles_12mo  int64
Flight_trans_12    int64
Days_since_enroll  int64
Award?             int64
dtype: object
```

In [31]:
```
def norm_func(i):
    x = (i-i.min())/(i.max()-i.min())
    return(x)
```

In [32]:
```
df_norm = norm_func(data.iloc[:,1:])
df_norm
```

Out[32]:

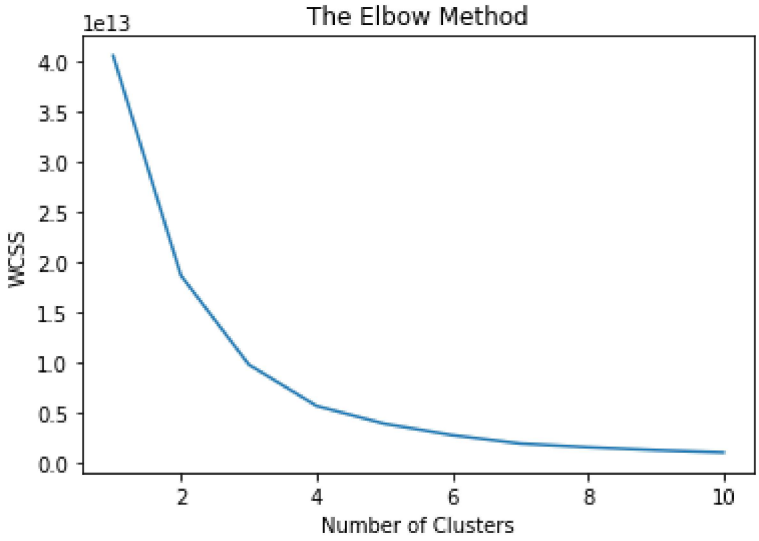
	Balance	Qual_miles	cc1_miles	cc2_miles	cc3_miles	Bonus_miles	Bonus_trans	Flight_m
0	0.016508	0.0	0.00	0.0	0.0	0.000660	0.011628	
1	0.011288	0.0	0.00	0.0	0.0	0.000815	0.023256	
2	0.024257	0.0	0.00	0.0	0.0	0.015636	0.046512	
3	0.008667	0.0	0.00	0.0	0.0	0.001896	0.011628	
4	0.057338	0.0	0.75	0.0	0.0	0.164211	0.302326	
...	...	...	...	...	...	...	...	
3994	0.010837	0.0	0.00	0.0	0.0	0.032330	0.046512	
3995	0.037766	0.0	0.00	0.0	0.0	0.003720	0.058140	
3996	0.043169	0.0	0.50	0.0	0.0	0.096505	0.093023	
3997	0.032202	0.0	0.00	0.0	0.0	0.001896	0.011628	
3998	0.001769	0.0	0.00	0.0	0.0	0.000000	0.000000	

3999 rows × 11 columns

In [33]:
```
X = data.iloc[:,[1,11]].values
```

In [34]:
```
wcss = []
for i in range(1,11):
    kmeans = KMeans(n_clusters=i,init = 'k-means++',random_state=0)
    kmeans.fit(X)
    wcss.append(kmeans.inertia_)
```

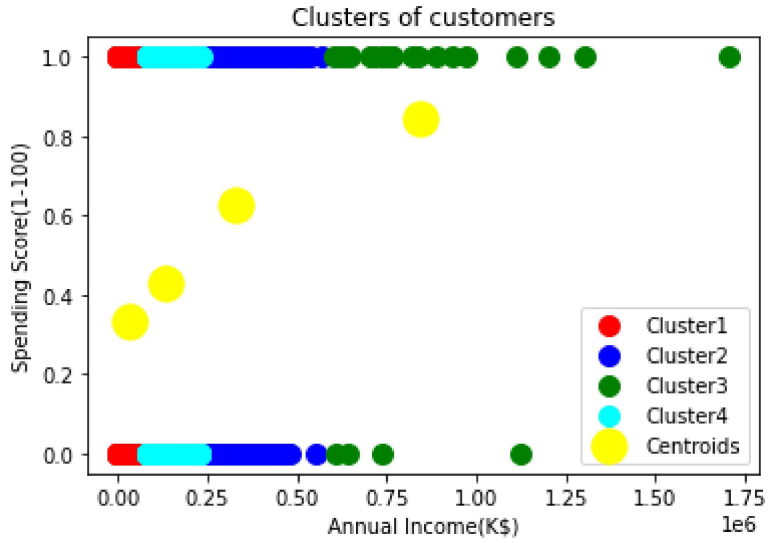
In [35]:
```
plt.plot(range(1,11),wcss)
plt.title('The Elbow Method')
plt.xlabel('Number of Clusters')
plt.ylabel('WCSS')
plt.show()
```



In [49]:
```
kmeans=KMeans(n_clusters=4,init='k-means++',random_state=0)
y_kmeans = kmeans.fit_predict(X)
y_kmeans
```

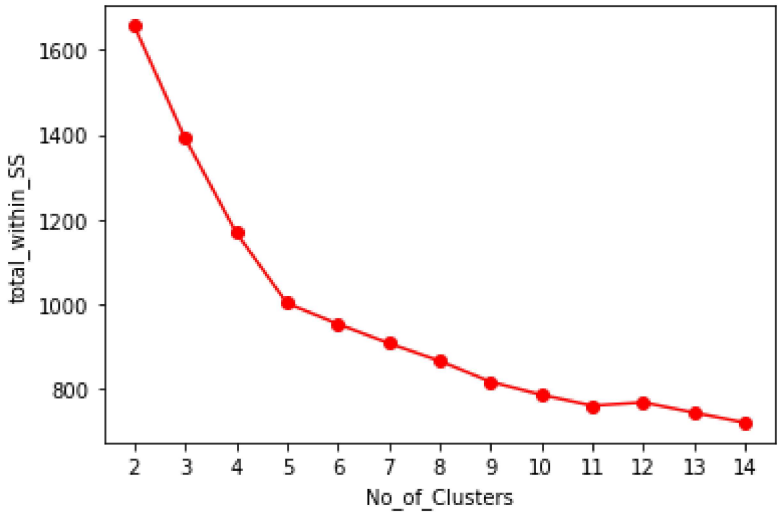
Out[49]:
```
array([0, 0, 0, ..., 0, 0, 0])
```

In [50]:
```
plt.scatter(X[y_kmeans==0,0],X[y_kmeans==0,1],s=100,c='red',label='Cluster1')
plt.scatter(X[y_kmeans==1,0],X[y_kmeans==1,1],s=100,c='blue',label='Cluster2')
plt.scatter(X[y_kmeans==2,0],X[y_kmeans==2,1],s=100,c='green',label='Cluster3')
plt.scatter(X[y_kmeans==3,0],X[y_kmeans==3,1],s=100,c='cyan',label='Cluster4')

plt.scatter(kmeans.cluster_centers_[:,0],kmeans.cluster_centers_[:,1],s=300,c='ye

plt.title('Clusters of customers')
plt.xlabel('Annual Income(K$)')
plt.ylabel('Spending Score(1-100)')
plt.legend()
plt.show()
```



In [55]:
```
k = list(range(2,15))
k
TWSS = [] # variable for storing total within sum of squares for each kmeans
for i in k:
    kmeans = KMeans(n_clusters = i)
    kmeans.fit(df_norm)
    WSS = [] # variable for storing within sum of squares for each cluster
    for j in range(i):
        WSS.append(sum(cdist(df_norm.iloc[kmeans.labels_==j,:],kmeans.cluster_cer
    TWSS.append(sum(WSS))
```

In [57]:
```
plt.plot(k,TWSS, 'ro-')
plt.xlabel("No_of_Clusters")
plt.ylabel("total_within_SS")
plt.xticks(k)
plt.show()
```



In [58]:
```
model = KMeans(n_clusters=5)
model.fit(df_norm)
```

Out[58]:
```
KMeans(n_clusters=5)
```

In [59]:
```
model.labels_
```

Out[59]:
```
array([0, 0, 0, ..., 1, 3, 3])
```

In [60]:
```
md = pd.Series(model.labels_)
```

In [61]:
```
data.columns
```

Out[61]:
```
Index(['ID#', 'Balance', 'Qual_miles', 'cc1_miles', 'cc2_miles', 'cc3_miles',
       'Bonus_miles', 'Bonus_trans', 'Flight_miles_12mo', 'Flight_trans_12',
       'Days_since_enroll', 'Award?'],
      dtype='object')
```

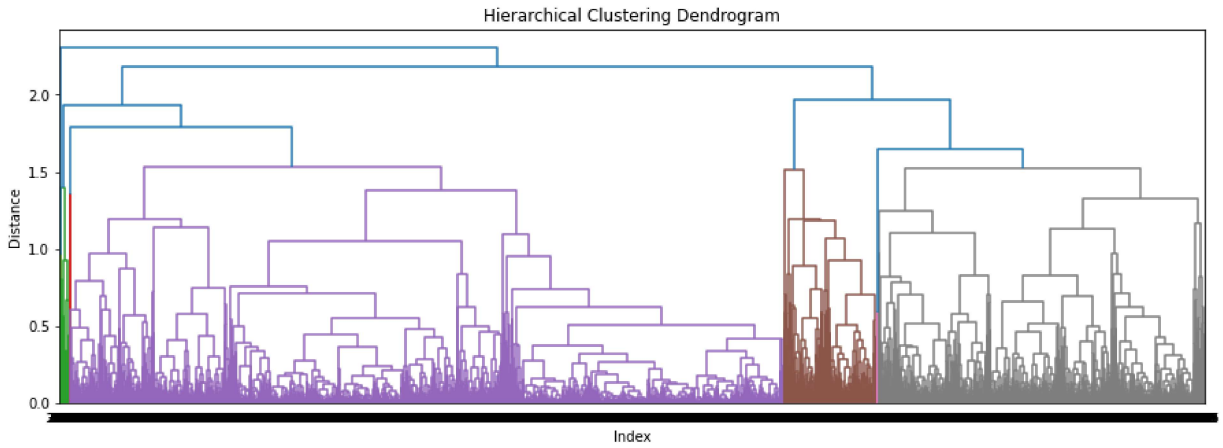
In [62]:
```
X = data[['Balance', 'Qual_miles','cc1_miles','cc2_miles','cc3_miles','Bonus_mile
clusters = KMeans(4)  # 4 clusters!
clusters.fit( X )
clusters.cluster_centers_
clusters.labels_

data['Crime_clusters'] = clusters.labels_
data.head()
data.sort_values(by=['Crime_clusters'],ascending = True)
X.head()
```

Out[62]:

	Balance	Qual_miles	cc1_miles	cc2_miles	cc3_miles	Bonus_miles	Bonus_trans	Flight_miles_
0	28143	0	1	1	1	174	1	
1	19244	0	1	1	1	215	2	
2	41354	0	1	1	1	4123	4	
3	14776	0	1	1	1	500	1	
4	97752	0	4	1	1	43300	26	

In [69]:
```
c = linkage(df_norm,method='complete',metric='euclidean')
```

In [72]:
```
plt.figure(figsize=(15, 5))
plt.title('Hierarchical Clustering Dendrogram')
plt.xlabel('Index')
plt.ylabel('Distance')
sch.dendrogram(
    c,
    leaf_rotation=0.,  # rotates the x axis labels
    leaf_font_size=8.,  # font size for the x axis labels
)
plt.show()
```



In [76]:
```
agc= AgglomerativeClustering(n_clusters=4,linkage='complete',affinity = "euclidea
```

In [77]:
```
agc.labels_
```

Out[77]:
```
array([0, 0, 0, ..., 2, 0, 0], dtype=int64)
```

In [78]:
```
cluster_labels = pd.Series(agc.labels_)
```

In [ ]:
```

```