AIRLINES -CLUSTERING

In [1]:
```
# load the libraies
import numpy as np
import pandas as pd
```

In [2]:
```
# read the data
data = pd.read_excel("EastWestAirlines.xlsx", sheet_name='data')
```

In [3]:
```
data.head()
```

Out[3]:

	ID#	Balance	Qual_miles	cc1_miles	cc2_miles	cc3_miles	Bonus_miles	Bonus_trans	Flight_miles_12mo	Flight_trans_12	Days_since_enrol
0	1	28143	0	1	1	1	174	1	0	0	700
1	2	19244	0	1	1	1	215	2	0	0	696
2	3	41354	0	1	1	1	4123	4	0	0	703
3	4	14776	0	1	1	1	500	1	0	0	695
4	5	97752	0	4	1	1	43300	26	2077	4	693

EDA

In [4]:
```
# do the eda part
data.describe()
```

Out[4]:

	ID#	Balance	Qual_miles	cc1_miles	cc2_miles	cc3_miles	Bonus_miles	Bonus_trans	Flight_miles_12mo	Flight
count	3999.000000	3.999000e+03	3999.000000	3999.000000	3999.000000	3999.000000	3999.000000	3999.00000	3999.000000	399
mean	2014.819455	7.360133e+04	144.114529	2.059515	1.014504	1.012253	17144.846212	11.60190	460.055764	
std	1160.764358	1.007757e+05	773.663804	1.376919	0.147650	0.195241	24150.967826	9.60381	1400.209171	
min	1.000000	0.000000e+00	0.000000	1.000000	1.000000	1.000000	0.000000	0.00000	0.000000	
25%	1010.500000	1.852750e+04	0.000000	1.000000	1.000000	1.000000	1250.000000	3.00000	0.000000	
50%	2016.000000	4.309700e+04	0.000000	1.000000	1.000000	1.000000	7171.000000	12.00000	0.000000	
75%	3020.500000	9.240400e+04	0.000000	3.000000	1.000000	1.000000	23800.500000	17.00000	311.000000	
max	4021.000000	1.704838e+06	11148.000000	5.000000	3.000000	5.000000	263685.000000	86.00000	30817.000000	

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3999 entries, 0 to 3998
Data columns (total 12 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   ID#                3999 non-null   int64
 1   Balance            3999 non-null   int64
 2   Qual_miles         3999 non-null   int64
 3   cc1_miles          3999 non-null   int64
 4   cc2_miles          3999 non-null   int64
 5   cc3_miles          3999 non-null   int64
 6   Bonus_miles        3999 non-null   int64
 7   Bonus_trans        3999 non-null   int64
 8   Flight_miles_12mo  3999 non-null   int64
 9   Flight_trans_12    3999 non-null   int64
 10  Days_since_enroll  3999 non-null   int64
 11  Award?             3999 non-null   int64
dtypes: int64(12)
memory usage: 375.0 KB
```

In [6]:
```
data.isnull().sum()
```

Out[6]:
```
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
```

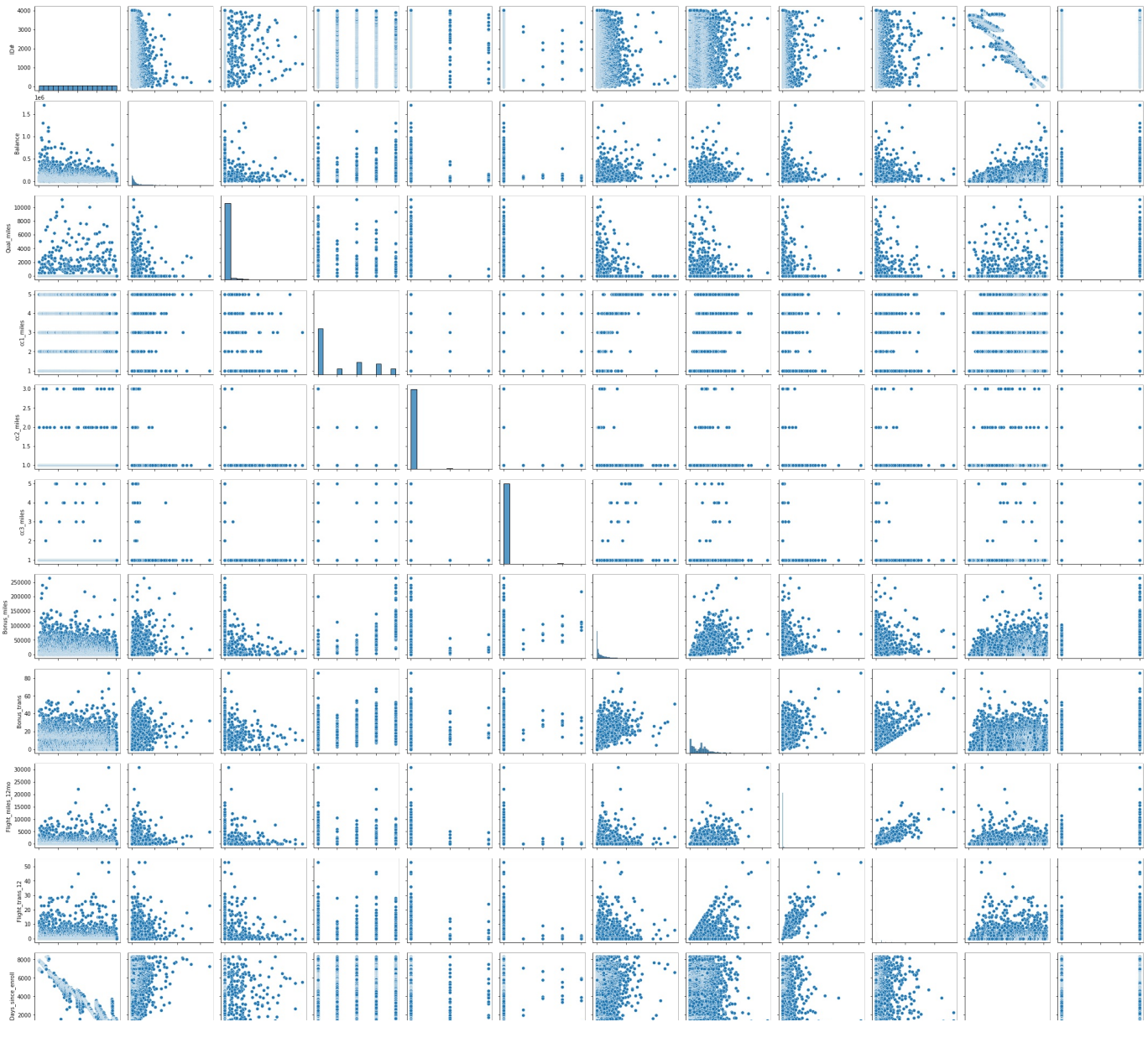
In [7]:
```
data.duplicated()   # there are no duplicates
```

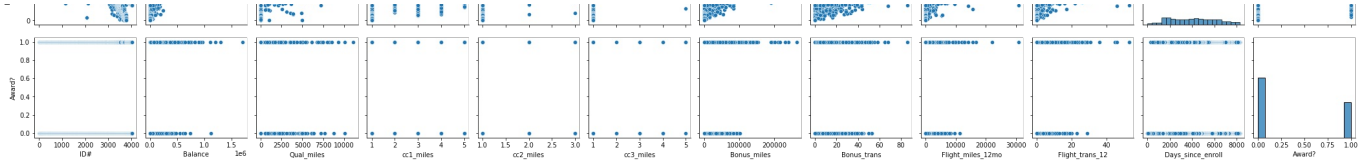
Out[7]:
```
0       False
1       False
2       False
3       False
4       False
        ...
3994    False
3995    False
3996    False
3997    False
3998    False
Length: 3999, dtype: bool
```

In [8]:
```
import seaborn as sns
```

In [9]:
```
sns.pairplot(data)
```

Out[9]:
```
<seaborn.axisgrid.PairGrid at 0x18441cdd220>
```





In [10]:
```
air = data.drop("ID#",axis=1)   # drop the unncessary cloumns
```

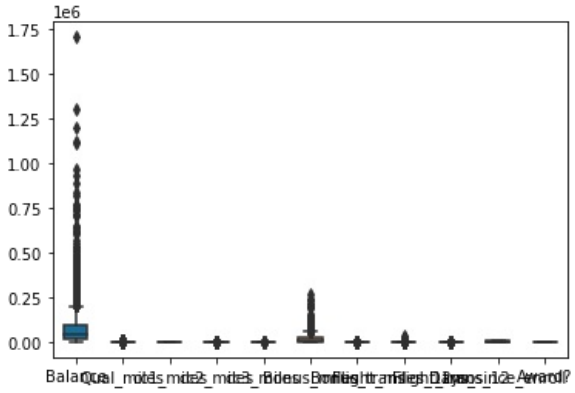
In [11]:
```
air.head()
```

Out[11]:

	Balance	Qual_miles	cc1_miles	cc2_miles	cc3_miles	Bonus_miles	Bonus_trans	Flight_miles_12mo	Flight_trans_12	Days_since_enroll	Aw
0	28143	0	1	1	1	174	1	0	0	7000	
1	19244	0	1	1	1	215	2	0	0	6968	
2	41354	0	1	1	1	4123	4	0	0	7034	
3	14776	0	1	1	1	500	1	0	0	6952	
4	97752	0	4	1	1	43300	26	2077	4	6935	

In [12]:
```
sns.boxplot(data=air,width=0.5,palette = "colorblind")   # see the outliers
```

Out[12]:
```
<AxesSubplot:>
```

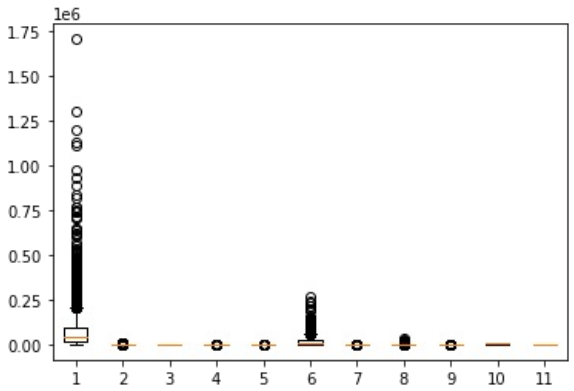


In [13]:
```
import matplotlib.pyplot as plt
%matplotlib inline
```

In [14]:
```
plt.boxplot(air)
```

Out[14]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x1844bab8130>,
  <matplotlib.lines.Line2D at 0x1844bab84c0>,
  <matplotlib.lines.Line2D at 0x1844bac3a60>,
  <matplotlib.lines.Line2D at 0x1844bac3df0>,
  <matplotlib.lines.Line2D at 0x1844badc3d0>,
  <matplotlib.lines.Line2D at 0x1844badc760>,
  <matplotlib.lines.Line2D at 0x1844bae5d00>,
  <matplotlib.lines.Line2D at 0x1844baf00d0>,
  <matplotlib.lines.Line2D at 0x1844bafb670>,
  <matplotlib.lines.Line2D at 0x1844bafba00>,
  <matplotlib.lines.Line2D at 0x1844bb07fa0>,
  <matplotlib.lines.Line2D at 0x1844bb12370>,
  <matplotlib.lines.Line2D at 0x1844bb1c910>,
  <matplotlib.lines.Line2D at 0x1844bb1cca0>,
  <matplotlib.lines.Line2D at 0x1844bb35280>,
  <matplotlib.lines.Line2D at 0x1844bb35610>,
  <matplotlib.lines.Line2D at 0x1844bb3fbb0>,
  <matplotlib.lines.Line2D at 0x1844bb3ff40>,
  <matplotlib.lines.Line2D at 0x1844bb55520>,
  <matplotlib.lines.Line2D at 0x1844bb558b0>,
  <matplotlib.lines.Line2D at 0x1844bb61e50>,
  <matplotlib.lines.Line2D at 0x1844bb6d220>],
 'caps': [<matplotlib.lines.Line2D at 0x1844bab8850>,
  <matplotlib.lines.Line2D at 0x1844bab8be0>,
  <matplotlib.lines.Line2D at 0x1844bacf1c0>,
  <matplotlib.lines.Line2D at 0x1844bacf550>,
  <matplotlib.lines.Line2D at 0x1844badcaf0>,
```

```
 <matplotlib.lines.Line2D at 0x1844badce80>,
 <matplotlib.lines.Line2D at 0x1844baf0460>,
 <matplotlib.lines.Line2D at 0x1844baf07f0>,
 <matplotlib.lines.Line2D at 0x1844bafbd90>,
 <matplotlib.lines.Line2D at 0x1844bb07160>,
 <matplotlib.lines.Line2D at 0x1844bb12700>,
 <matplotlib.lines.Line2D at 0x1844bb12a90>,
 <matplotlib.lines.Line2D at 0x1844bb26070>,
 <matplotlib.lines.Line2D at 0x1844bb26400>,
 <matplotlib.lines.Line2D at 0x1844bb359a0>,
 <matplotlib.lines.Line2D at 0x1844bb35d30>,
 <matplotlib.lines.Line2D at 0x1844bb4a310>,
 <matplotlib.lines.Line2D at 0x1844bb4a6a0>,
 <matplotlib.lines.Line2D at 0x1844bb55c40>,
 <matplotlib.lines.Line2D at 0x1844bb55fd0>,
 <matplotlib.lines.Line2D at 0x1844bb6d5b0>,
 <matplotlib.lines.Line2D at 0x1844bb6d940>],
'boxes': [<matplotlib.lines.Line2D at 0x1844baa8d60>,
 <matplotlib.lines.Line2D at 0x1844bac36d0>,
 <matplotlib.lines.Line2D at 0x1844badc040>,
 <matplotlib.lines.Line2D at 0x1844bae5970>,
 <matplotlib.lines.Line2D at 0x1844bafb2e0>,
 <matplotlib.lines.Line2D at 0x1844bb07c10>,
 <matplotlib.lines.Line2D at 0x1844bb1c580>,
 <matplotlib.lines.Line2D at 0x1844bb26eb0>,
 <matplotlib.lines.Line2D at 0x1844bb3f820>,
 <matplotlib.lines.Line2D at 0x1844bb55190>,
 <matplotlib.lines.Line2D at 0x1844bb61ac0>],
'medians': [<matplotlib.lines.Line2D at 0x1844bab8f70>,
 <matplotlib.lines.Line2D at 0x1844bacf8e0>,
 <matplotlib.lines.Line2D at 0x1844bae5250>,
 <matplotlib.lines.Line2D at 0x1844baf0b80>,
 <matplotlib.lines.Line2D at 0x1844bb074f0>,
 <matplotlib.lines.Line2D at 0x1844bb12e20>,
 <matplotlib.lines.Line2D at 0x1844bb26790>,
 <matplotlib.lines.Line2D at 0x1844bb3f100>,
 <matplotlib.lines.Line2D at 0x1844bb4aa30>,
 <matplotlib.lines.Line2D at 0x1844bb613a0>,
 <matplotlib.lines.Line2D at 0x1844bb6dcd0>],
'fliers': [<matplotlib.lines.Line2D at 0x1844bac3340>,
 <matplotlib.lines.Line2D at 0x1844bacfc70>,
 <matplotlib.lines.Line2D at 0x1844bae55e0>,
 <matplotlib.lines.Line2D at 0x1844baf0f10>,
 <matplotlib.lines.Line2D at 0x1844bb07880>,
 <matplotlib.lines.Line2D at 0x1844bb1c1f0>,
 <matplotlib.lines.Line2D at 0x1844bb26b20>,
 <matplotlib.lines.Line2D at 0x1844bb3f490>,
 <matplotlib.lines.Line2D at 0x1844bb4adc0>,
 <matplotlib.lines.Line2D at 0x1844bb61730>,
 <matplotlib.lines.Line2D at 0x1844bb770a0>],
'means': []}
```



In [15]:
```
air.head()
```

Out[15]:

	Balance	Qual_miles	cc1_miles	cc2_miles	cc3_miles	Bonus_miles	Bonus_trans	Flight_miles_12mo	Flight_trans_12	Days_since_enroll	Aw
0	28143	0	1	1	1	174	1	0	0	7000	
1	19244	0	1	1	1	215	2	0	0	6968	
2	41354	0	1	1	1	4123	4	0	0	7034	
3	14776	0	1	1	1	500	1	0	0	6952	
4	97752	0	4	1	1	43300	26	2077	4	6935	

Normalise the Data

Normalise the Data

In [16]:
```
# Normalization function
def air_data(i):
    x = (i-i.min())/(i.max()-i.min())
    return (x)
```

In [17]:
```
# Normalized data frame (considering the numerical part of data)
df_norm = air_data(air.iloc[:,:])
```

In [18]:
```
df_norm
```

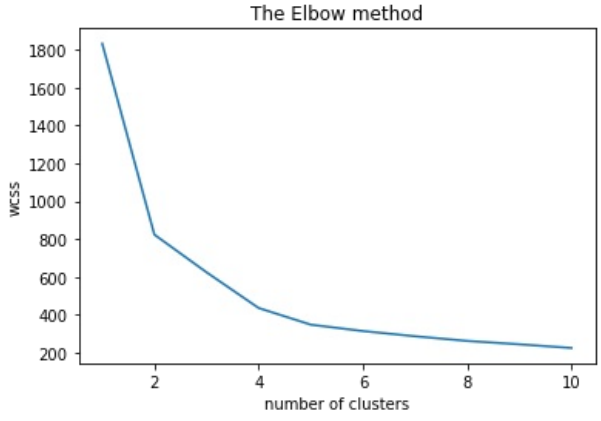
Out[18]:

	Balance	Qual_miles	cc1_miles	cc2_miles	cc3_miles	Bonus_miles	Bonus_trans	Flight_miles_12mo	Flight_trans_12	Days_since_enroll
0	0.016508	0.0	0.00	0.0	0.0	0.000660	0.011628	0.000000	0.000000	0.843742
1	0.011288	0.0	0.00	0.0	0.0	0.000815	0.023256	0.000000	0.000000	0.839884
2	0.024257	0.0	0.00	0.0	0.0	0.015636	0.046512	0.000000	0.000000	0.847842
3	0.008667	0.0	0.00	0.0	0.0	0.001896	0.011628	0.000000	0.000000	0.837955
4	0.057338	0.0	0.75	0.0	0.0	0.164211	0.302326	0.067398	0.075472	0.835905
...	...	...	...	...	...	...	...	...	...	...
3994	0.010837	0.0	0.00	0.0	0.0	0.032330	0.046512	0.006490	0.018868	0.168917
3995	0.037766	0.0	0.00	0.0	0.0	0.003720	0.058140	0.000000	0.000000	0.167953
3996	0.043169	0.0	0.50	0.0	0.0	0.096505	0.093023	0.000000	0.000000	0.168797
3997	0.032202	0.0	0.00	0.0	0.0	0.001896	0.011628	0.016225	0.018868	0.168676
3998	0.001769	0.0	0.00	0.0	0.0	0.000000	0.000000	0.000000	0.000000	0.168314

3999 rows × 11 columns

WCSS - within cluster Sum of Square

In [19]:
```
# We are Using Elbow Graph to find optimum number of clusters (K value) from K values range
from sklearn.cluster import KMeans
wcss = []
for i in range (1,11):
    kmeans = KMeans(n_clusters = i, init = 'k-means++', random_state= 42)
    kmeans.fit(df_norm)
    wcss.append(kmeans.inertia_)
plt.plot(range(1,11),wcss)
plt.title('The Elbow method')
plt.xlabel('number of clusters')
plt.ylabel('wcss')
plt.show()
```



In [20]:
```
# by the elbow there are 4cluster
```

In [21]:
```
wcss
```

Out[21]:
```
[1830.793212858415,
 823.6756984125229,
```

```
625.1684881570748,
436.70885761932624,
348.9433217254146,
315.3155964842897,
287.7310054422432,
263.2132188914902,
245.40617070458336,
226.1081210825213]
```

K-Means

In [22]:
```
# Build The Cluster Algorithm With K =4
kmeans = KMeans(n_clusters=4, init='k-means++', random_state=0)
y_means = kmeans.fit_predict(df_norm)
```

In [23]:
```
y_means
```

Out[23]:
```
array([1, 1, 1, ..., 2, 1, 1])
```

In [24]:
```
#assign clusters
```

In [25]:
```
df_norm['cluster'] = y_means
df_norm.head()
```

Out[25]:

	Balance	Qual_miles	cc1_miles	cc2_miles	cc3_miles	Bonus_miles	Bonus_trans	Flight_miles_12mo	Flight_trans_12	Days_since_enroll	Av
0	0.016508	0.0	0.00	0.0	0.0	0.000660	0.011628	0.000000	0.000000	0.843742	
1	0.011288	0.0	0.00	0.0	0.0	0.000815	0.023256	0.000000	0.000000	0.839884	
2	0.024257	0.0	0.00	0.0	0.0	0.015636	0.046512	0.000000	0.000000	0.847842	
3	0.008667	0.0	0.00	0.0	0.0	0.001896	0.011628	0.000000	0.000000	0.837955	
4	0.057338	0.0	0.75	0.0	0.0	0.164211	0.302326	0.067398	0.075472	0.835905	

In [26]:
```
import sklearn.cluster as cluster
```

In [27]:
```
from sklearn import metrics
```

In [28]:
```
# Using The Silhoutee Score we can whether k=4 cluster are not
for i in range(3,13):
    labels=cluster.KMeans(n_clusters=i,init="k-means++",random_state=200).fit(df_norm).labels_
    print ("Silhouette score for k(clusters) = "+str(i)+" is "
           +str(metrics.silhouette_score(df_norm,labels,metric="euclidean",sample_size=1000,random_state=200)))
```

```
Silhouette score for k(clusters) = 3 is 0.6480378208016775
Silhouette score for k(clusters) = 4 is 0.726233789529253
Silhouette score for k(clusters) = 5 is 0.5799835846035146
Silhouette score for k(clusters) = 6 is 0.5022817704861466
Silhouette score for k(clusters) = 7 is 0.4469641282049033
Silhouette score for k(clusters) = 8 is 0.3910623108235298
Silhouette score for k(clusters) = 9 is 0.338984637761425
Silhouette score for k(clusters) = 10 is 0.34630934069459757
Silhouette score for k(clusters) = 11 is 0.3468986945826195
Silhouette score for k(clusters) = 12 is 0.3453761520875926
```

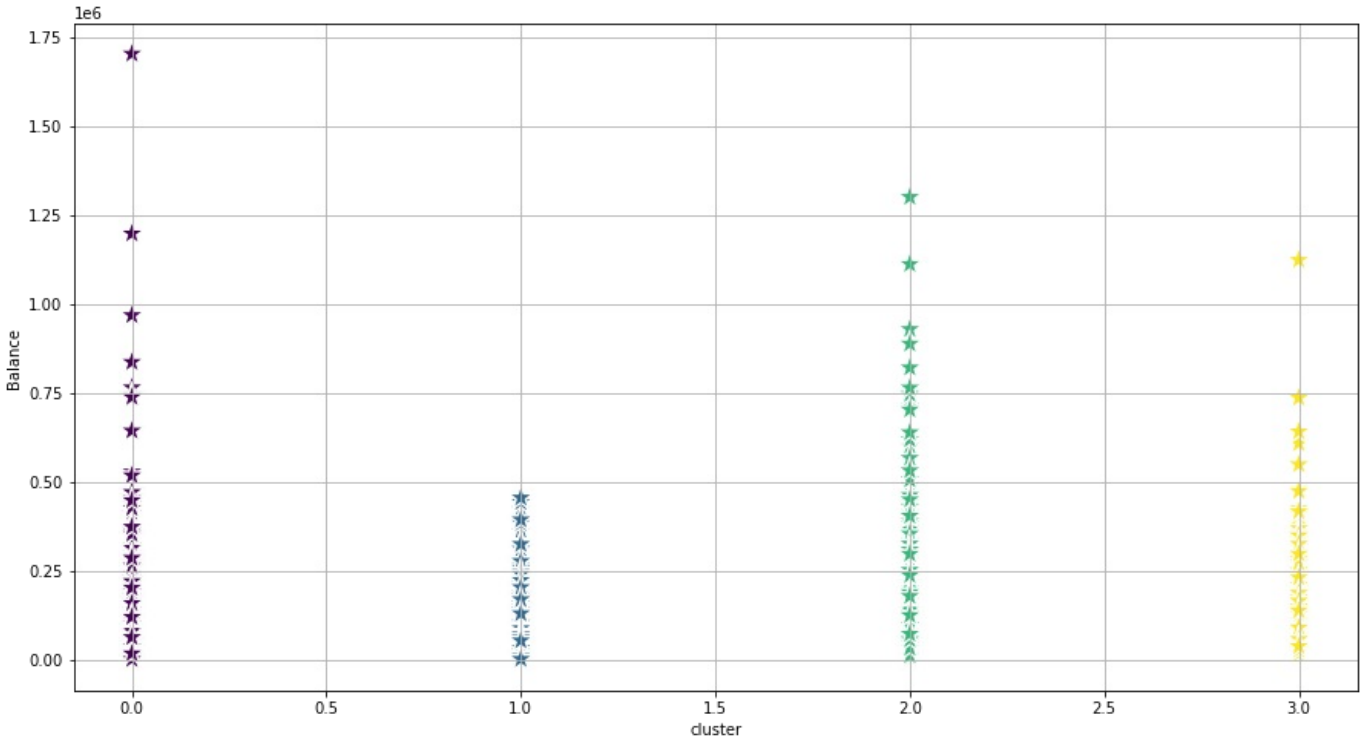
In [29]:
```
model=KMeans(n_clusters=4)
model.fit(df_norm)
model.labels_
```

Out[29]:
```
array([1, 1, 1, ..., 2, 1, 1])
```

Plot the Clusters

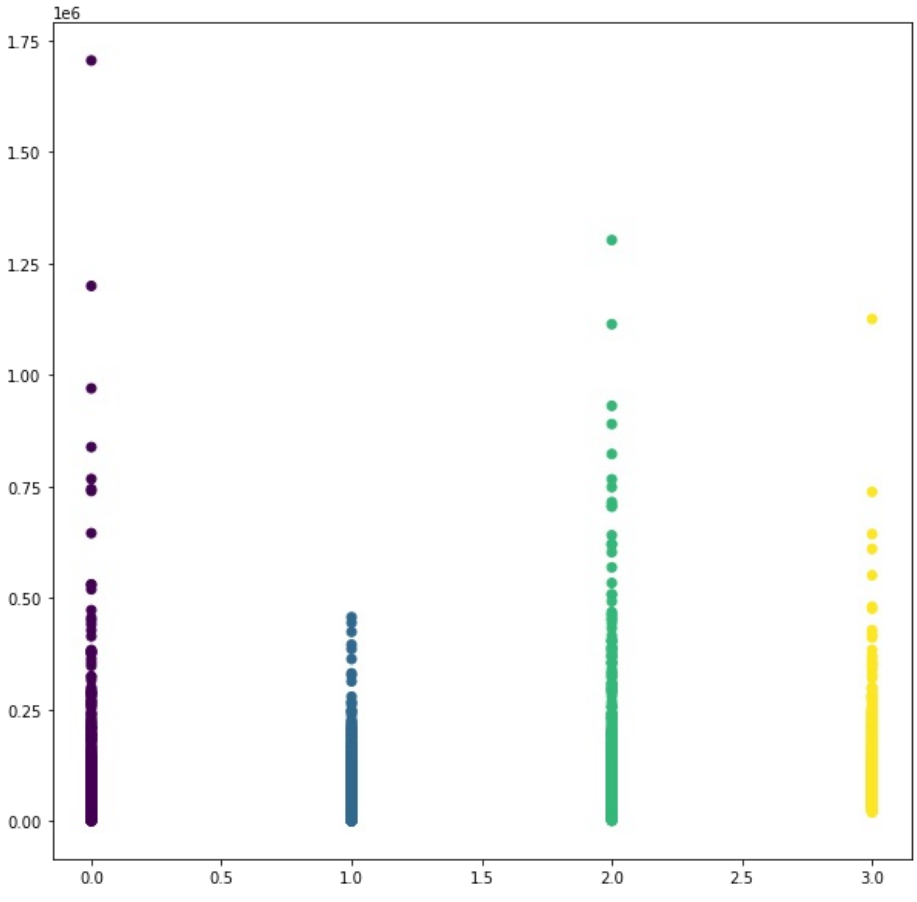
In [30]:
```
plt.figure(figsize=(15,8))
sns.scatterplot(df_norm['cluster'],data['Balance'],c=kmeans.labels_,s=300,marker='*')
plt.grid()
plt.show();
```

```
C:\Users\rajesh\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable
s as keyword args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```



In [31]:
```
plt.figure(figsize=(10, 10))
plt.scatter(df_norm['cluster'],data['Balance'], c=kmeans.labels_)
```

Out[31]:
```
<matplotlib.collections.PathCollection at 0x1843e161760>
```



Hiearchical-Clustering

In [32]:
```
air.head()
```

Out[32]:

	Balance	Qual_miles	cc1_miles	cc2_miles	cc3_miles	Bonus_miles	Bonus_trans	Flight_miles_12mo	Flight_trans_12	Days_since_enroll	Aw
0	28143	0	1	1	1	174	1	0	0	7000	
1	19244	0	1	1	1	215	2	0	0	6968	
2	41354	0	1	1	1	4123	4	0	0	7034	
3	14776	0	1	1	1	500	1	0	0	6952	
4	97752	0	4	1	1	43300	26	2077	4	6935	

In [33]:
```
df_norm.head()
```

Out[33]:

	Balance	Qual_miles	cc1_miles	cc2_miles	cc3_miles	Bonus_miles	Bonus_trans	Flight_miles_12mo	Flight_trans_12	Days_since_enroll	A
0	0.016508	0.0	0.00	0.0	0.0	0.000660	0.011628	0.000000	0.000000	0.843742	
1	0.011288	0.0	0.00	0.0	0.0	0.000815	0.023256	0.000000	0.000000	0.839884	
2	0.024257	0.0	0.00	0.0	0.0	0.015636	0.046512	0.000000	0.000000	0.847842	
3	0.008667	0.0	0.00	0.0	0.0	0.001896	0.011628	0.000000	0.000000	0.837955	
4	0.057338	0.0	0.75	0.0	0.0	0.164211	0.302326	0.067398	0.075472	0.835905	

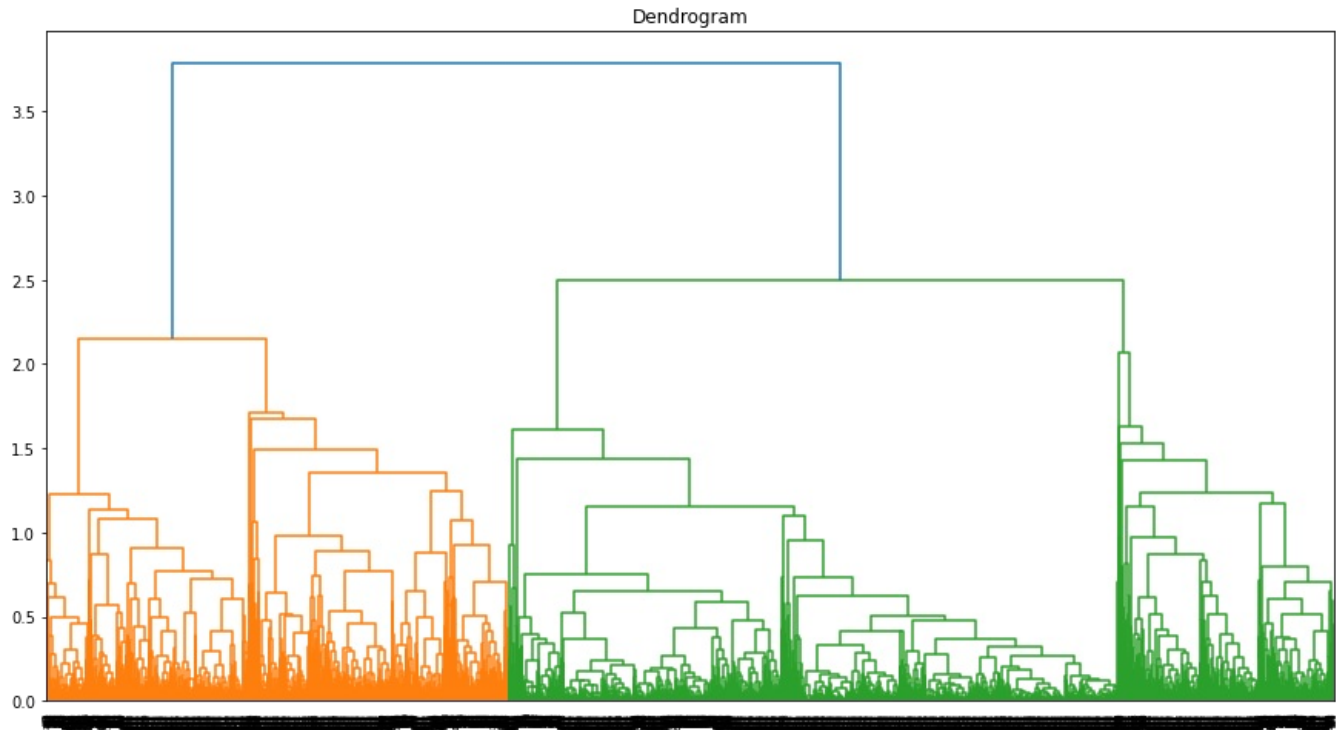
DenDrogram

In [34]:
```
import scipy.cluster.hierarchy as sch
```

In [35]:
```
from sklearn.cluster import AgglomerativeClustering
```

In [36]:
```
# create the dendrogram
plt.figure(figsize=(15,8))
dendrogram = sch.dendrogram(sch.linkage(df_norm,method='complete'))
plt.title('Dendrogram')
```

Out[36]:
```
Text(0.5, 1.0, 'Dendrogram')
```



Train Model

In [37]:
```
hc = AgglomerativeClustering(n_clusters=4, affinity = 'euclidean',linkage = 'complete')
```

In [38]:
```
hc
```

Out[38]:
```
AgglomerativeClustering(linkage='complete', n_clusters=4)
```

In [39]:
```
y_hc = hc.fit_predict(df_norm)
```

In [40]:
```
y_hc
```

Out[40]:
```
array([2, 2, 2, ..., 1, 2, 2], dtype=int64)
```

In [41]:
```
y=pd.DataFrame(hc.fit_predict(df_norm),columns=['clustersid'])
y['clustersid'].value_counts()
```

Out[41]:
```
2    1891
1     808
0     673
3     627
Name: clustersid, dtype: int64
```

In [42]:
```
df_norm['clustersid']=hc.labels_
df_norm.head(10)
```

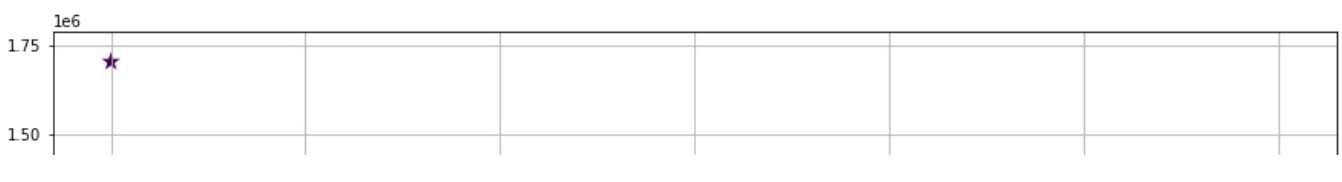
Out[42]:

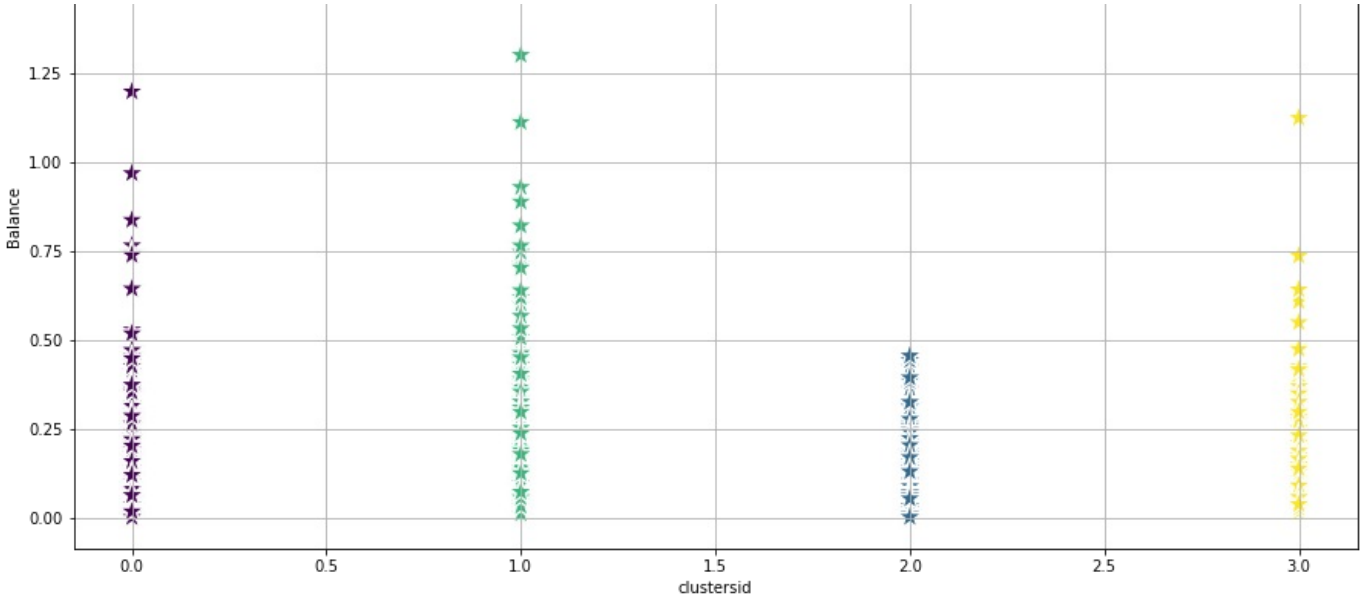
	Balance	Qual_miles	cc1_miles	cc2_miles	cc3_miles	Bonus_miles	Bonus_trans	Flight_miles_12mo	Flight_trans_12	Days_since_enroll	Av
0	0.016508	0.0	0.00	0.0	0.0	0.000660	0.011628	0.000000	0.000000	0.843742	
1	0.011288	0.0	0.00	0.0	0.0	0.000815	0.023256	0.000000	0.000000	0.839884	
2	0.024257	0.0	0.00	0.0	0.0	0.015636	0.046512	0.000000	0.000000	0.847842	
3	0.008667	0.0	0.00	0.0	0.0	0.001896	0.011628	0.000000	0.000000	0.837955	
4	0.057338	0.0	0.75	0.0	0.0	0.164211	0.302326	0.067398	0.075472	0.835905	
5	0.009631	0.0	0.00	0.0	0.0	0.000000	0.000000	0.000000	0.000000	0.836749	
6	0.049808	0.0	0.50	0.0	0.0	0.104223	0.290698	0.000000	0.000000	0.843019	
7	0.012233	0.0	0.00	0.0	0.0	0.019910	0.046512	0.008112	0.018868	0.836267	
8	0.259850	0.0	0.50	0.5	0.0	0.006648	0.500000	0.124931	0.226415	0.837473	
9	0.061507	0.0	0.50	0.0	0.0	0.107803	0.325581	0.037317	0.056604	0.835423	

Plot Clusters

In [43]:
```
plt.figure(figsize=(15,8))
sns.scatterplot(df_norm['clustersid'],data['Balance'],c=kmeans.labels_,s=300,marker='*')
plt.grid()
plt.show();
```

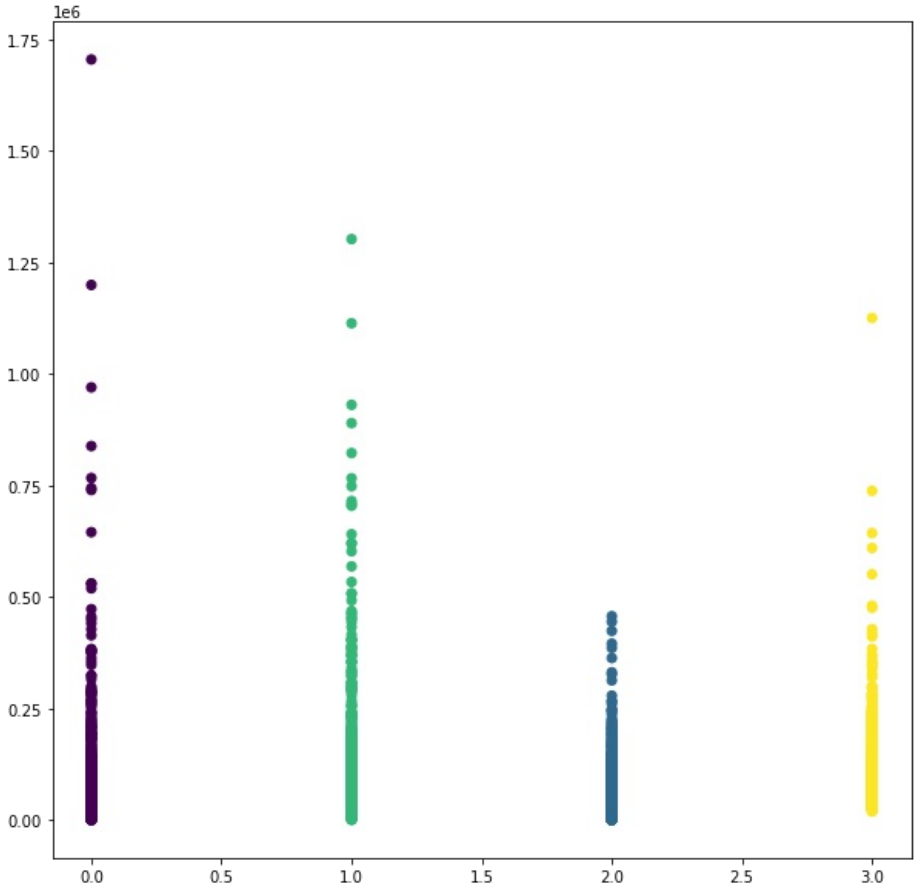
```
C:\Users\rajesh\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable
s as keyword args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```





In [44]:
```
plt.figure(figsize=(10, 10))
plt.scatter(df_norm['clustersid'],data['Balance'], c=kmeans.labels_)
```

Out[44]:
```
<matplotlib.collections.PathCollection at 0x18454135400>
```



DB-SCAN

Density-Based Spatial Clustering Applications with Noise (DB-scan)

In [45]:
```
from sklearn.cluster import DBSCAN
```

In [46]:
```
dbscan=DBSCAN(eps=1,min_samples=4)
dbscan.fit(df_norm)
```

Out[46]:
```
DBSCAN(eps=1, min_samples=4)
```

In [47]:
```
dbscan.labels_
```

Out[47]:
```
array([0, 0, 0, ..., 1, 0, 0], dtype=int64)
```

In [48]:
```
df_norm['clusters']=dbscan.labels_
df_norm.head()
```

Out[48]:

	Balance	Qual_miles	cc1_miles	cc2_miles	cc3_miles	Bonus_miles	Bonus_trans	Flight_miles_12mo	Flight_trans_12	Days_since_enroll	Av
0	0.016508	0.0	0.00	0.0	0.0	0.000660	0.011628	0.000000	0.000000	0.843742	
1	0.011288	0.0	0.00	0.0	0.0	0.000815	0.023256	0.000000	0.000000	0.839884	
2	0.024257	0.0	0.00	0.0	0.0	0.015636	0.046512	0.000000	0.000000	0.847842	
3	0.008667	0.0	0.00	0.0	0.0	0.001896	0.011628	0.000000	0.000000	0.837955	
4	0.057338	0.0	0.75	0.0	0.0	0.164211	0.302326	0.067398	0.075472	0.835905	

In [49]:
```
df_norm.groupby('cluster').agg(['mean']).reset_index()
```

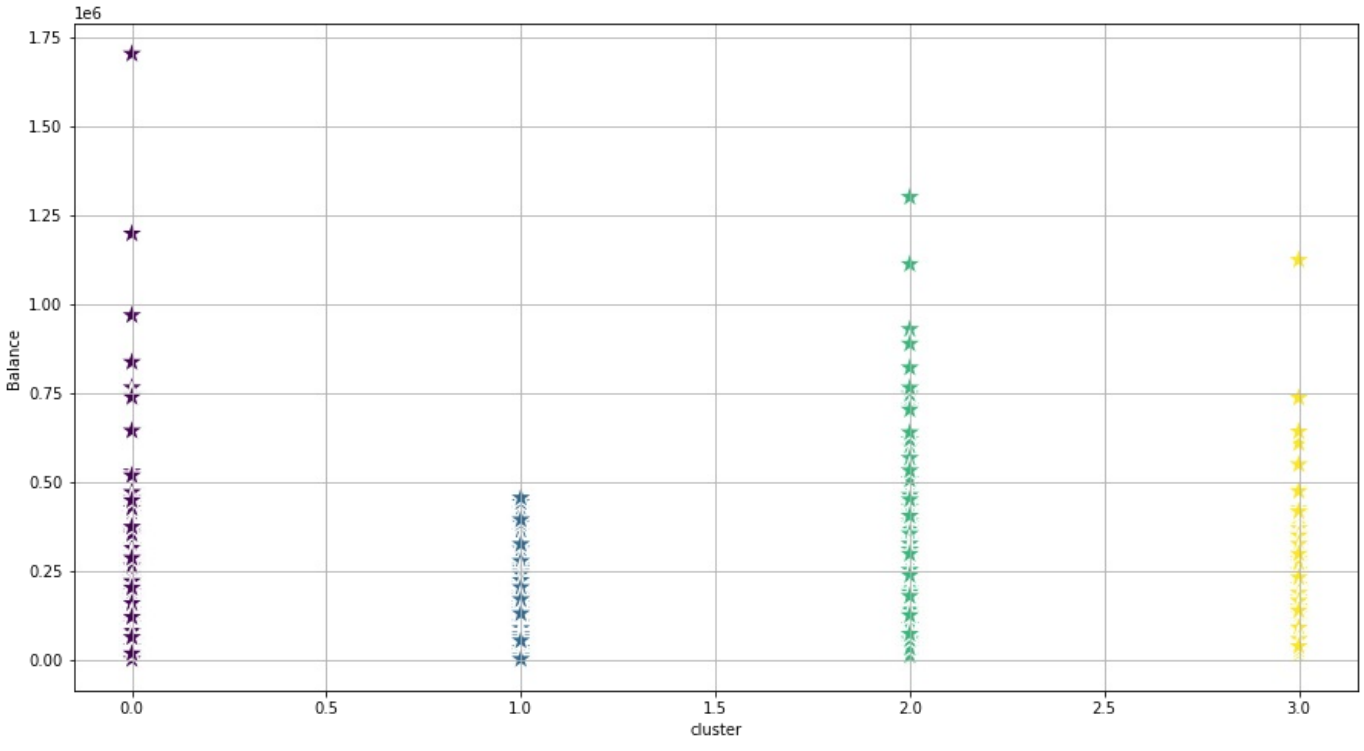
Out[49]:

	cluster	Balance	Qual_miles	cc1_miles	cc2_miles	cc3_miles	Bonus_miles	Bonus_trans	Flight_miles_12mo	Flight_trans_12	Days_since_e
		mean	mean	mean	mean	mean	mean	mean	mean	mean	
0	0	0.048995	0.026054	0.039004	0.016345	0.002229	0.033564	0.121825	0.033427	0.059407	0.52
1	1	0.023768	0.008071	0.022475	0.008990	0.001058	0.012575	0.075154	0.007357	0.012432	0.43
2	2	0.063535	0.017791	0.728960	0.000619	0.006498	0.172970	0.234904	0.023160	0.040421	0.58
3	3	0.069201	0.007215	0.640351	0.000797	0.005582	0.117843	0.200289	0.007302	0.011947	0.53

In [50]:
```
import matplotlib.pyplot as plt
%matplotlib inline
```

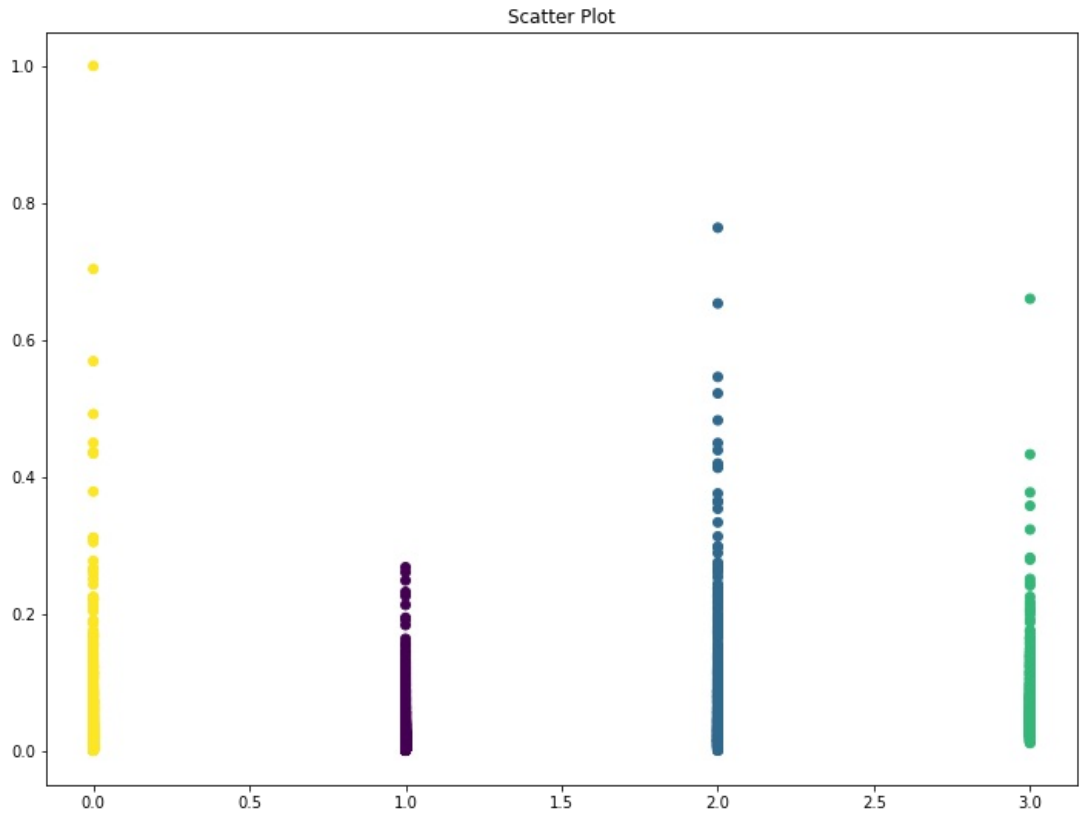
In [51]:
```
plt.figure(figsize=(15,8))
sns.scatterplot(df_norm['cluster'],data['Balance'],c=kmeans.labels_,s=300,marker='*')
plt.grid()
plt.show();
```

```
C:\Users\rajesh\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable
s as keyword args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```



In [52]:
```
plt.figure(figsize=(12,9))
plt.title('Scatter Plot')
plt.scatter(df_norm['cluster'],df_norm['Balance'], c=dbscan.labels_)
```

Out[52]:
```
<matplotlib.collections.PathCollection at 0x18453fbae80>
```



There Are Four Clusters Formed

In [ ]:
```

```