Performing K-Means clustering method on Airline Data set

```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
from sklearn.cluster import KMeans
```

```
data = pd.read_excel('/content/EastWestAirlines.xlsx', sheet_name='data')
```

```
data.head()
```

	ID#	Balance	Qual_miles	cc1_miles	cc2_miles	cc3_miles	Bonus_miles	Bonus_trai
0	1	28143	0	1	1	1	174	
1	2	19244	0	1	1	1	215	
2	3	41354	0	1	1	1	4123	
3	4	14776	0	1	1	1	500	
4	5	97752	0	4	1	1	43300	2

```
data.isna().sum()
```

```
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
```

```
#Normalization function
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaled_data_df = scaler.fit_transform(data.iloc[:,1:])
```

```
scaled_data_df
```

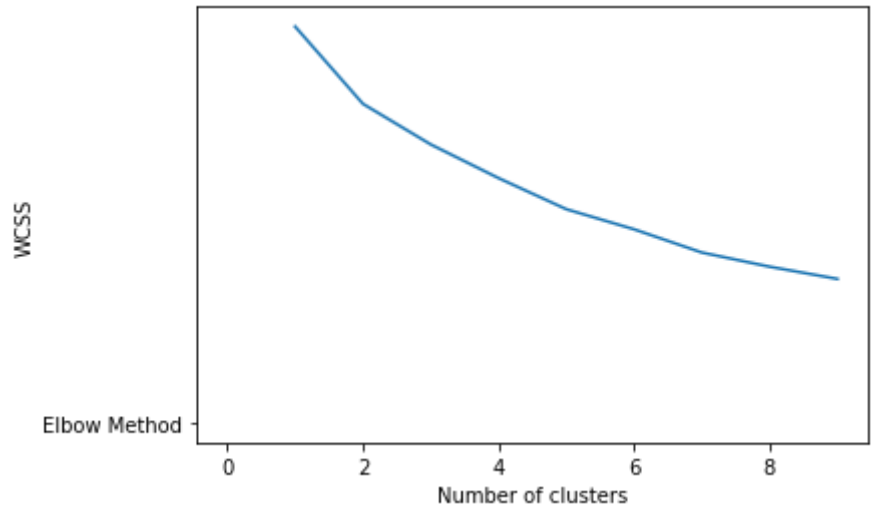
```
array([[-4.51140783e-01, -1.86298687e-01, -7.69578406e-01, ...,
        -3.62167870e-01,  1.39545434e+00, -7.66919299e-01],
       [-5.39456874e-01, -1.86298687e-01, -7.69578406e-01, ...,
```

```
       -3.62167870e-01,  1.37995704e+00, -7.66919299e-01],
      [-3.20031232e-01, -1.86298687e-01, -7.69578406e-01, ...,
       -3.62167870e-01,  1.41192021e+00, -7.66919299e-01],
      ...,
      [-4.29480975e-05, -1.86298687e-01,  6.83121167e-01, ...,
       -3.62167870e-01, -1.31560393e+00,  1.30391816e+00],
      [-1.85606976e-01, -1.86298687e-01, -7.69578406e-01, ...,
       -9.85033311e-02, -1.31608822e+00, -7.66919299e-01],
      [-7.00507951e-01, -1.86298687e-01, -7.69578406e-01, ...,
       -3.62167870e-01, -1.31754109e+00, -7.66919299e-01]])
```

```
#How to find optimum number of cluster
# The K-Means algorithm aims to choose centroids that minimise the inertia, of (WCSS) with
```

```
wcss = []
for i in range(1,10):
    kmeans = KMeans(n_clusters=i, random_state=0)
    kmeans.fit(scaled_data_df)
    wcss.append(kmeans.inertia_)

plt.plot(range(1,10), wcss)
plt.plot('Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()
```



K = 5 is the good value to be considered

```
#Build cluster algorithm
from  sklearn.cluster import KMeans
cluster_new = KMeans(5, random_state=42)
cluster_new.fit(scaled_data_df)
```

```
     KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
            n_clusters=5, n_init=10, n_jobs=None, precompute_distances='auto',
            random_state=42, tol=0.0001, verbose=0)
```

```
cluster_new.labels_
```

```
array([1, 1, 1, ..., 2, 1, 1], dtype=int32)
```

```
#Assign clusters to the data set
data['Clusterid_new'] = cluster_new.labels_
```

```
data.head()
```

	ID#	Balance	Qual_miles	cc1_miles	cc2_miles	cc3_miles	Bonus_miles	Bonus_t
0	1	28143	0	1	1	1	174	
1	2	19244	0	1	1	1	215	
2	3	41354	0	1	1	1	4123	
3	4	14776	0	1	1	1	500	
4	5	97752	0	4	1	1	43300	

```
data.groupby('Clusterid_new').agg(['mean']).reset_index()
```

	Clusterid_new	ID#	Balance	Qual_miles	cc1_miles	cc2_miles	cc
		mean	mean	mean	mean	mean	me
0	0	2057.295082	119660.491803	5351.065574	2.000000	1.000000	
1	1	2237.521395	43543.198098	42.349049	1.225436	1.019414	
2	2	1598.158147	117485.166933	59.420128	3.707668	1.002396	
3	3	1757.802721	190251.952381	458.734694	2.224490	1.040816	
4	4	1664.866667	138061.400000	78.800000	3.466667	1.000000	

✓ 1s completed at 2:28 AM