In [1]:
```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
from sklearn.metrics.pairwise import cosine_similarity
```

In [3]:
```python
!pip install kneed
```

```
Collecting kneed
  Downloading kneed-0.8.5-py3-none-any.whl.metadata (5.5 kB)
Requirement already satisfied: numpy>=1.14.2 in c:\users\legion\anaconda3\lib\site-packages (from kneed) (1.26.4)
Requirement already satisfied: scipy>=1.0.0 in c:\users\legion\anaconda3\lib\site-packages (from kneed) (1.13.1)
Downloading kneed-0.8.5-py3-none-any.whl (10 kB)
Installing collected packages: kneed
Successfully installed kneed-0.8.5
```

In [5]:
```python
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
from sklearn.cluster import DBSCAN
from sklearn.cluster import AgglomerativeClustering
import matplotlib.pyplot as plt
from kneed import KneeLocator
from sklearn.metrics import silhouette_score
import seaborn as sns
```

In [7]:
```python
inpt1=pd.read_csv('CC GENERAL.csv')
inpt1.head()
```

Out[7]:

	CUST_ID	BALANCE	BALANCE_FREQUENCY	PURCHASES	ONEOFF_PURCHASES	INSTALLMENTS_PURCHASES	CASH_ADVANCE	PURCHASES_FREQUENCY	ONEOFF_PURCHASES_FREQUENCY	PURCHASES_INSTALLMENTS_FREQUENCY	CASH_ADVANCE_FREQUENCY	CASH_ADVANCE_TRX	PURCHASES_
0	C10001	40.900749	0.818182	95.40	0.00	95.4	0.000000	0.166667	0.000000	0.083333	0.000000	0	
1	C10002	3202.467416	0.909091	0.00	0.00	0.0	6442.945483	0.000000	0.000000	0.000000	0.250000	4	
2	C10003	2495.148862	1.000000	773.17	773.17	0.0	0.000000	1.000000	1.000000	0.000000	0.000000	0	
3	C10004	1666.670542	0.636364	1499.00	1499.00	0.0	205.788017	0.083333	0.083333	0.000000	0.083333	1	
4	C10005	817.714335	1.000000	16.00	16.00	0.0	0.000000	0.083333	0.083333	0.000000	0.000000	0	

In [9]:
```python
inpt1.describe()
```

Out[9]:

	BALANCE	BALANCE_FREQUENCY	PURCHASES	ONEOFF_PURCHASES	INSTALLMENTS_PURCHASES	CASH_ADVANCE	PURCHASES_FREQUENCY	ONEOFF_PURCHASES_FREQUENCY	PURCHASES_INSTALLMENTS_FREQUENCY	CASH_ADVANCE_FREQUENCY	CASH_ADVANCE_TRX	PURCHASES_TRX
count	8950.000000	8950.000000	8950.000000	8950.000000	8950.000000	8950.000000	8950.000000	8950.000000	8950.000000	8950.000000	8950.000000	8950.000000
mean	1564.474828	0.877271	1003.204834	592.437371	411.067645	978.871112	0.490351	0.202458	0.364437	0.135144	3.248827	14.709832
std	2081.531879	0.236904	2136.634782	1659.887917	904.338115	2097.163877	0.401371	0.298336	0.397448	0.200121	6.824647	24.857649
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	128.281915	0.888889	39.635000	0.000000	0.000000	0.000000	0.083333	0.000000	0.000000	0.000000	0.000000	1.000000
50%	873.385231	1.000000	361.280000	38.000000	89.000000	0.000000	0.500000	0.083333	0.166667	0.000000	0.000000	7.000000
75%	2054.140036	1.000000	1110.130000	577.405000	468.637500	1113.821139	0.916667	0.300000	0.750000	0.222222	4.000000	17.000000
max	19043.138560	1.000000	49039.570000	40761.250000	22500.000000	47137.211760	1.000000	1.000000	1.000000	1.500000	123.000000	358.000000

In [11]:
```python
inpt1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8950 entries, 0 to 8949
Data columns (total 18 columns):
 #   Column                            Non-Null Count  Dtype
---  ------                            --------------  -----
 0   CUST_ID                           8950 non-null   object
 1   BALANCE                           8950 non-null   float64
 2   BALANCE_FREQUENCY                 8950 non-null   float64
 3   PURCHASES                         8950 non-null   float64
 4   ONEOFF_PURCHASES                  8950 non-null   float64
 5   INSTALLMENTS_PURCHASES            8950 non-null   float64
 6   CASH_ADVANCE                      8950 non-null   float64
 7   PURCHASES_FREQUENCY               8950 non-null   float64
 8   ONEOFF_PURCHASES_FREQUENCY        8950 non-null   float64
 9   PURCHASES_INSTALLMENTS_FREQUENCY  8950 non-null   float64
 10  CASH_ADVANCE_FREQUENCY            8950 non-null   float64
 11  CASH_ADVANCE_TRX                  8950 non-null   int64
 12  PURCHASES_TRX                     8950 non-null   int64
 13  CREDIT_LIMIT                      8949 non-null   float64
 14  PAYMENTS                          8950 non-null   float64
 15  MINIMUM_PAYMENTS                  8637 non-null   float64
 16  PRC_FULL_PAYMENT                  8950 non-null   float64
 17  TENURE                            8950 non-null   int64
dtypes: float64(14), int64(3), object(1)
memory usage: 1.2+ MB
```

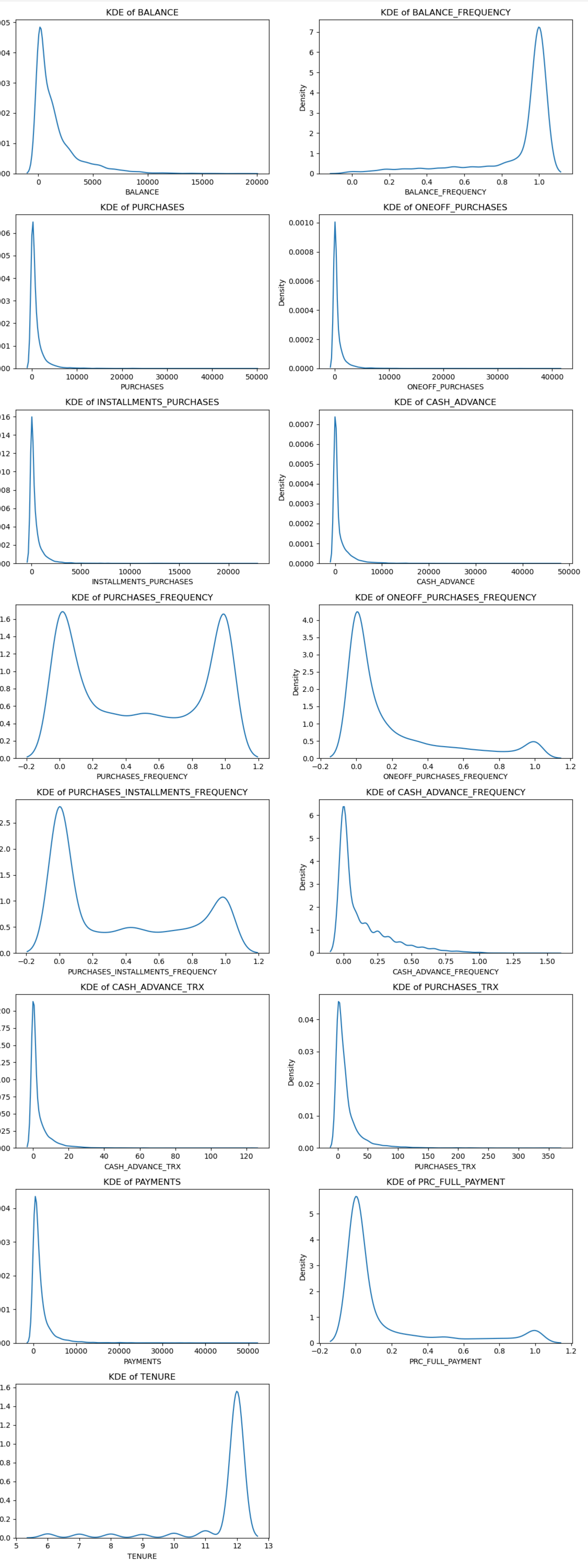
In [13]:
```python
inpt1.columns
```

Out[13]:
```
Index(['CUST_ID', 'BALANCE', 'BALANCE_FREQUENCY', 'PURCHASES',
       'ONEOFF_PURCHASES', 'INSTALLMENTS_PURCHASES', 'CASH_ADVANCE',
       'PURCHASES_FREQUENCY', 'ONEOFF_PURCHASES_FREQUENCY',
       'PURCHASES_INSTALLMENTS_FREQUENCY', 'CASH_ADVANCE_FREQUENCY',
       'CASH_ADVANCE_TRX', 'PURCHASES_TRX', 'CREDIT_LIMIT', 'PAYMENTS',
       'MINIMUM_PAYMENTS', 'PRC_FULL_PAYMENT', 'TENURE'],
      dtype='object')
```

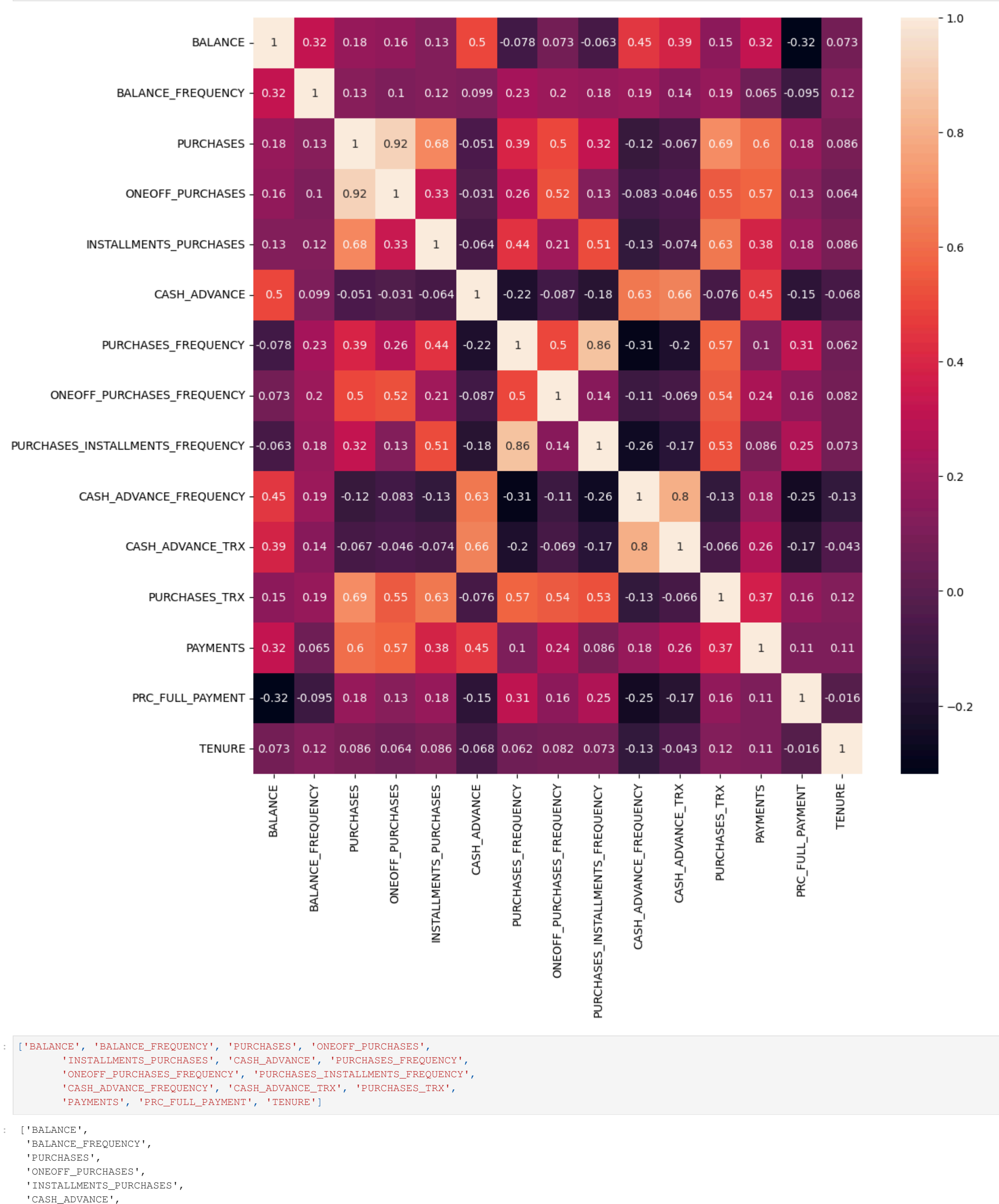
In [15]:
```python
inpt1.drop('CREDIT_LIMIT', axis=1, inplace=True)
inpt1.drop('MINIMUM_PAYMENTS', axis=1, inplace=True)
```

In [17]:
```python
inpt1.drop('CUST_ID', axis=1, inplace=True)
```

In [19]:
```python
numeric_columns = inpt1.select_dtypes(exclude='object').columns
num_plots = len(numeric_columns)

fig, axes = plt.subplots(nrows=(num_plots // 2) + (num_plots % 2), ncols=2, figsize=(12, num_plots * 2))


axes = axes.flatten()


for i, cl in enumerate(numeric_columns):
    sns.kdeplot(inpt1[cl], ax=axes[i])
    axes[i].set_title(f"KDE of {cl}")


for i in range(len(numeric_columns), len(axes)):
    fig.delaxes(axes[i])

plt.tight_layout()
plt.show()
```



In [25]:
```python
plt.figure(figsize=(12,12))
sns.heatmap(inpt1.corr(), annot=True)
plt.show()
```



In [29]:
```python
['BALANCE', 'BALANCE_FREQUENCY', 'PURCHASES', 'ONEOFF_PURCHASES',
       'INSTALLMENTS_PURCHASES', 'CASH_ADVANCE', 'PURCHASES_FREQUENCY',
       'ONEOFF_PURCHASES_FREQUENCY', 'PURCHASES_INSTALLMENTS_FREQUENCY',
       'CASH_ADVANCE_FREQUENCY', 'CASH_ADVANCE_TRX', 'PURCHASES_TRX',
       'PAYMENTS', 'PRC_FULL_PAYMENT', 'TENURE']
```

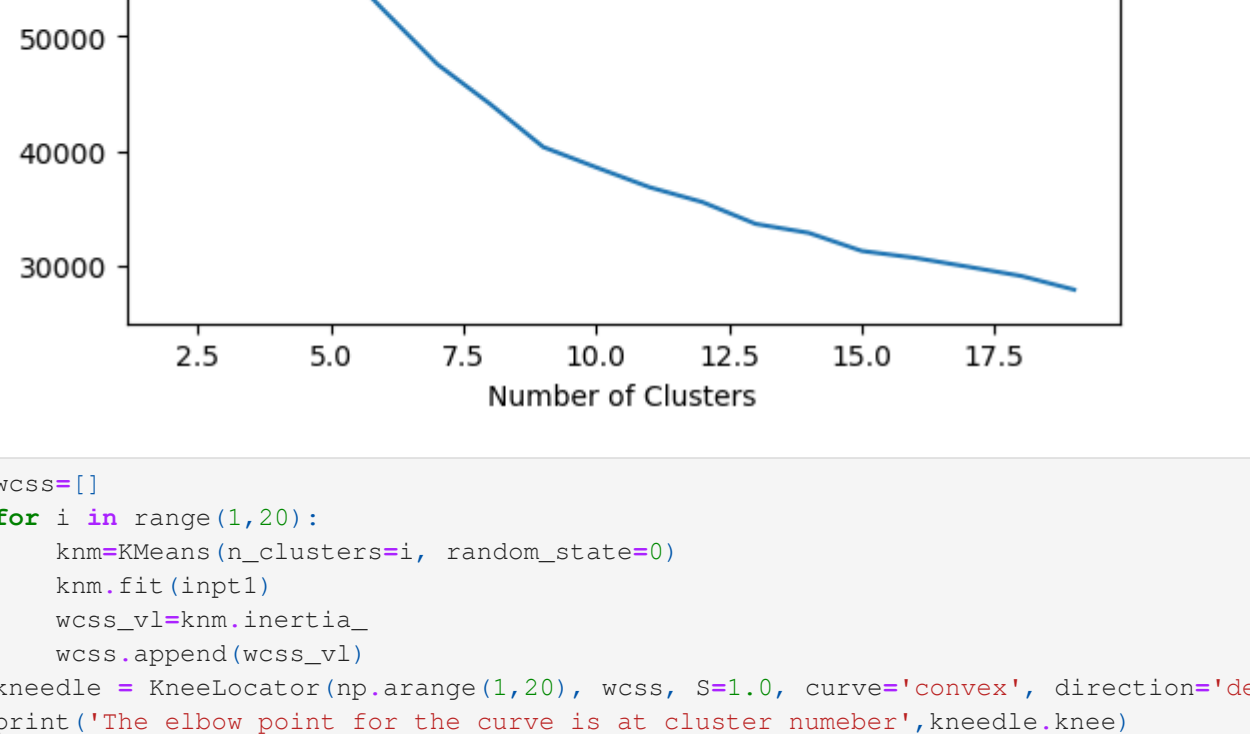
Out[29]:
```
['BALANCE',
 'BALANCE_FREQUENCY',
 'PURCHASES',
 'ONEOFF_PURCHASES',
 'INSTALLMENTS_PURCHASES',
 'CASH_ADVANCE',
 'PURCHASES_FREQUENCY',
 'ONEOFF_PURCHASES_FREQUENCY',
 'PURCHASES_INSTALLMENTS_FREQUENCY',
 'CASH_ADVANCE_FREQUENCY',
 'CASH_ADVANCE_TRX',
 'PURCHASES_TRX',
 'PAYMENTS',
 'PRC_FULL_PAYMENT',
 'TENURE']
```

In [31]:
```python
inpt1.drop(['PURCHASES_TRX','INSTALLMENTS_PURCHASES','ONEOFF_PURCHASES','PURCHASES_FREQUENCY'], axis=1, inplace=True)
```

In [33]:
```python
ss=StandardScaler()
ss.fit(inpt1)
inpt1=ss.transform(inpt1)
```

In [35]:
```python
wcss=[]
for i in range(2,20):
    knm=KMeans(n_clusters=i, random_state=0)
    knm.fit(inpt1)
    wcss_vl=knm.inertia_
    wcss.append(wcss_vl)
    print("Silhouette Coefficient for num_cluster",i, silhouette_score(inpt1, knm.labels_))
plt.xlabel('Number of Clusters')
plt.ylabel('WCSS')
plt.plot(np.arange(2,20), wcss)
plt.show()
```

```
Silhouette Coefficient for num_cluster 2 0.2369075172817349
Silhouette Coefficient for num_cluster 3 0.24850284374974113
Silhouette Coefficient for num_cluster 4 0.22401437761285276
Silhouette Coefficient for num_cluster 5 0.24153899681208704
Silhouette Coefficient for num_cluster 6 0.2546902784595453
Silhouette Coefficient for num_cluster 7 0.23779886202560196
Silhouette Coefficient for num_cluster 8 0.23655037156370806
Silhouette Coefficient for num_cluster 9 0.23727202972318467
Silhouette Coefficient for num_cluster 10 0.22782302856961653
Silhouette Coefficient for num_cluster 11 0.22814757082687545
Silhouette Coefficient for num_cluster 12 0.22719352132148068
Silhouette Coefficient for num_cluster 13 0.23756089622772983
Silhouette Coefficient for num_cluster 14 0.2042885603540345
Silhouette Coefficient for num_cluster 15 0.24619980094076313
Silhouette Coefficient for num_cluster 16 0.2346060495268308
Silhouette Coefficient for num_cluster 17 0.20256235221361077
Silhouette Coefficient for num_cluster 18 0.2064924984315432
Silhouette Coefficient for num_cluster 19 0.21591913993299539
```



In [36]:
```python
wcss=[]
for i in range(1,20):
    knm=KMeans(n_clusters=i, random_state=0)
    knm.fit(inpt1)
    wcss_vl=knm.inertia_
    wcss.append(wcss_vl)
kneedle = KneeLocator(np.arange(1,20), wcss, S=1.0, curve='convex', direction='decreasing')
print('The elbow point for the curve is at cluster numeber',kneedle.knee)
```

```
The elbow point for the curve is at cluster numeber 7
```

In [37]:
```python
eps=np.arange(0.1, 0.8, 0.1)
min_samples=np.arange(5,13,1)
for ep, mns in zip(eps, min_samples):
    dbs=DBSCAN(eps=ep, min_samples=mns)
    dbs.fit(inpt1)
    print("Silhouette Coefficient for eps and min_samples",ep,mns,silhouette_score(inpt1, dbs.labels_))
```

```
Silhouette Coefficient for eps and min_samples 0.1 5 -0.44855180890108776
Silhouette Coefficient for eps and min_samples 0.2 6 -0.44163356737966186
Silhouette Coefficient for eps and min_samples 0.30000000000000004 7 -0.4040376377129612
Silhouette Coefficient for eps and min_samples 0.4 8 -0.41969069294801736
Silhouette Coefficient for eps and min_samples 0.5 9 -0.40236231372563974
```

```
Silhouette Coefficient for eps and min_samples 0.6 11 -0.31796253978781851
Silhouette Coefficient for eps and min_samples 0.7000000000000001 11 -0.31987380683393937
```