In [23]:
```
import pandas as pd
import numpy as np
import seaborn as sns
sns.set()
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt

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

In [6]:
```
credit = pd.read_csv('CC_GENERAL.csv')
credit
```

Out[6]:

	CUST_ID	BALANCE	BALANCE_FREQUENCY	PURCHASES	ONEOFF_PURCHASES	INSTALLMENTS_PUR
0	C10001	40.900749	0.818182	95.40	0.00	
1	C10002	3202.467416	0.909091	0.00	0.00	
2	C10003	2495.148862	1.000000	773.17	773.17	
3	C10004	1666.670542	0.636364	1499.00	1499.00	
4	C10005	817.714335	1.000000	16.00	16.00	
...	...	...	...	...	...	
8945	C19186	28.493517	1.000000	291.12	0.00	
8946	C19187	19.183215	1.000000	300.00	0.00	
8947	C19188	23.398673	0.833333	144.40	0.00	
8948	C19189	13.457564	0.833333	0.00	0.00	
8949	C19190	372.708075	0.666667	1093.25	1093.25	

8950 rows × 18 columns

In [7]:
```
credit.isna().sum()
```

Out[7]:
```
CUST_ID                             0
BALANCE                             0
BALANCE_FREQUENCY                   0
PURCHASES                           0
ONEOFF_PURCHASES                    0
INSTALLMENTS_PURCHASES              0
CASH_ADVANCE                        0
PURCHASES_FREQUENCY                 0
ONEOFF_PURCHASES_FREQUENCY          0
PURCHASES_INSTALLMENTS_FREQUENCY    0
CASH_ADVANCE_FREQUENCY              0
CASH_ADVANCE_TRX                    0
PURCHASES_TRX                       0
CREDIT_LIMIT                        1
PAYMENTS                            0
MINIMUM_PAYMENTS                  313
PRC_FULL_PAYMENT                    0
TENURE                              0
dtype: int64
```

In [8]:
```
mean_value=credit['CREDIT_LIMIT'].mean()
credit['CREDIT_LIMIT']=credit['CREDIT_LIMIT'].fillna(mean_value)
mean_value=credit['MINIMUM_PAYMENTS'].mean()
credit['MINIMUM_PAYMENTS']=credit['MINIMUM_PAYMENTS'].fillna(mean_value)
```

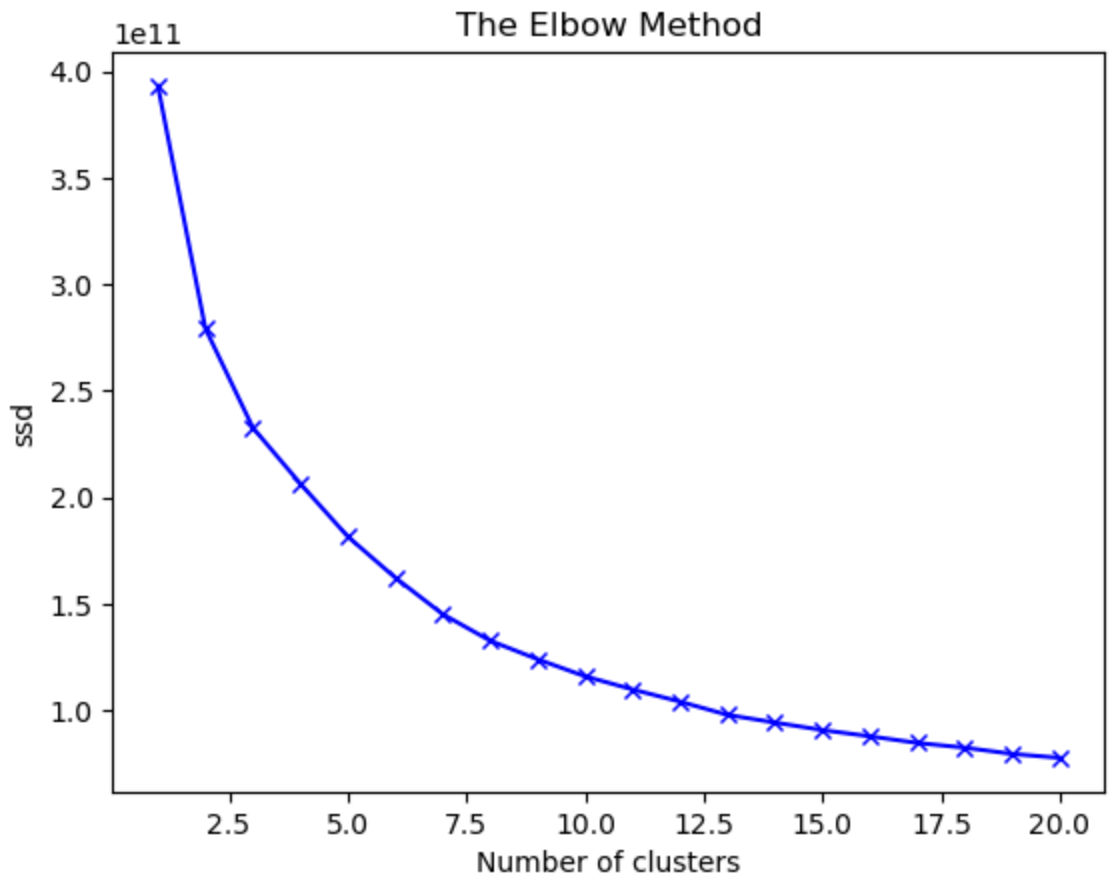
In [10]:
```
from sklearn.preprocessing import StandardScaler
df = credit.drop('CUST_ID', axis=1)
X = df.values[:,1:]
X = np.nan_to_num(X)
Clus_dataSet = StandardScaler().fit_transform(X)
Clus_dataSet
```

Out[10]:
```
array([[-0.24943448, -0.42489974, -0.35693402, ..., -0.31096755,
        -0.52555097,  0.36067954],
       [ 0.13432467, -0.46955188, -0.35693402, ...,  0.08931021,
         0.2342269 ,  0.36067954],
       [ 0.51808382, -0.10766823,  0.10888851, ..., -0.10166318,
        -0.52555097,  0.36067954],
       ...,
       [-0.18547673, -0.40196519, -0.35693402, ..., -0.33546549,
         0.32919999, -4.12276757],
       [-0.18547673, -0.46955188, -0.35693402, ..., -0.34690648,
         0.32919999, -4.12276757],
       [-0.88903307,  0.04214581,  0.30173158, ..., -0.33294642,
        -0.52555097, -4.12276757]])
```

In [11]:
```
from sklearn.cluster import KMeans
kmeans=KMeans(n_clusters=6,random_state =0)
kmeans.fit(df)
kmeans.labels_
```

Out[11]:
```
array([0, 3, 1, ..., 0, 0, 0])
```

In [15]:
```
from sklearn.cluster import KMeans
ssd = []
K=range(1,21)
for k in K:
    kmeans = KMeans(n_clusters = k)#, init = 'k-means++', random_state = 42)
    kmeans = kmeans.fit(df)
    ssd.append(kmeans.inertia_)
plt.plot(K, ssd,'bx-')
plt.title('The Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('ssd')
plt.show()
```



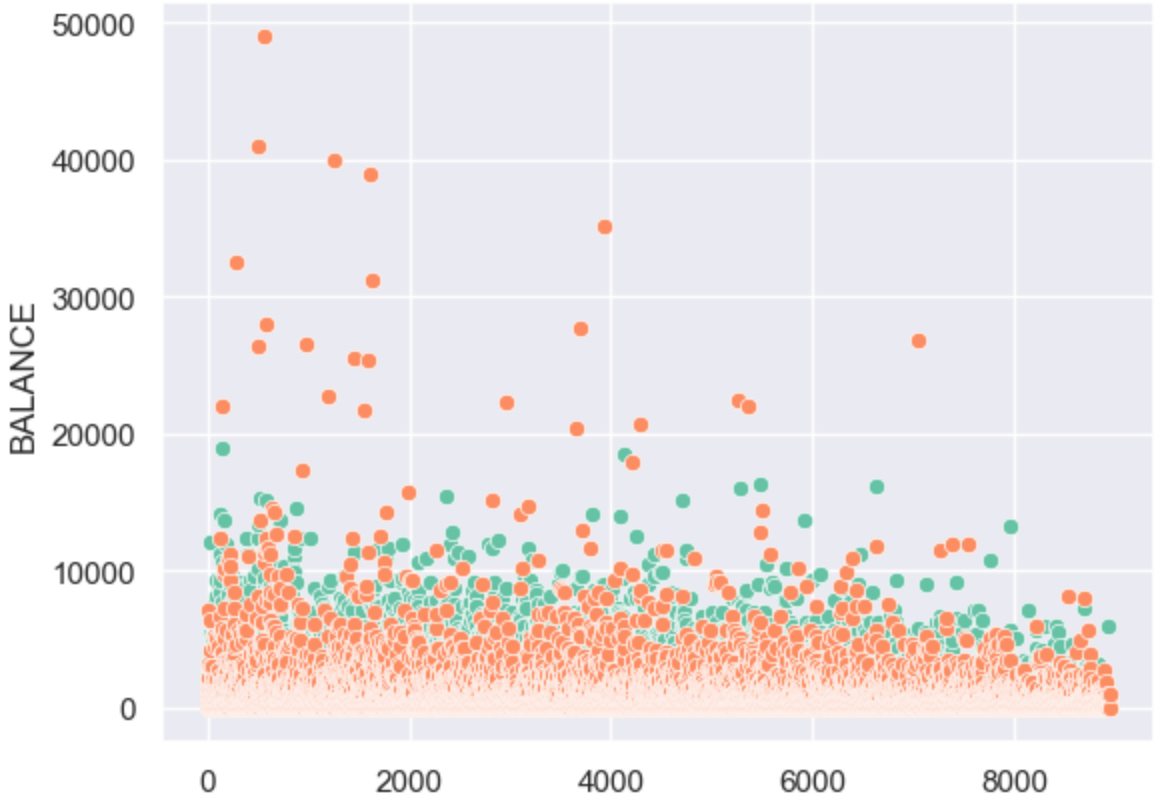
In [16]:
```
km_sample=KMeans(n_clusters=4)
km_sample.fit(df)
```

Out[16]:
```
KMeans(n_clusters=4)
```

In [18]:
```
labels_sample=km_sample.labels_
df['label']=labels_sample
```

In [35]:
```
sns.set_palette('Set2')
sns.scatterplot(df['BALANCE'],palette='Set1')
sns.scatterplot(df['PURCHASES'],palette='Set1')
```

Out[35]:
```
<Axes: ylabel='BALANCE'>
```



In [ ]: