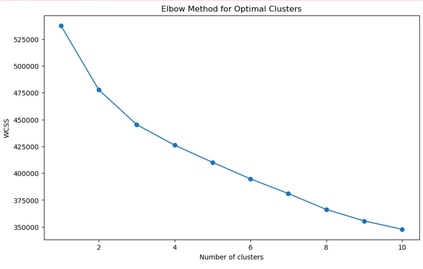
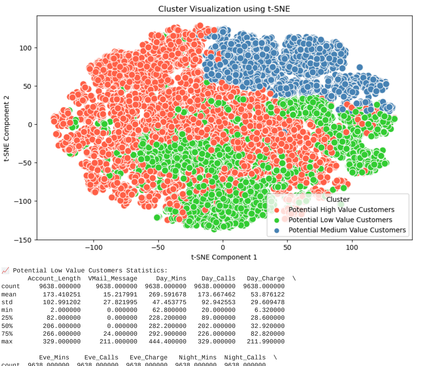
In [1]:
```
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
from sklearn.manifold import TSNE
import matplotlib.pyplot as plt
import seaborn as sns

# Load the dataset
file_path = r"C:\Users\manid\OneDrive\Documents\cleaned_churn dataset cdr.csv"
df = pd.read_csv(file_path)

# Ensure 'Churn' column is properly formatted
df['Churn'] = df['Churn'].astype(int)

# Select relevant numerical columns for clustering
numerical_cols = [
    'Account_Length', 'VMail_Message', 'Day_Mins', 'Day_Calls', 'Day_Charge',
    'Eve_Mins', 'Eve_Calls', 'Eve_Charge', 'Night_Mins', 'Night_Calls',
    'Night_Charge', 'Intl_Mins', 'Intl_Calls', 'Intl_Charge', 'CustServ_Calls'
]

X = df[numerical_cols]

# Standardize the features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Determine the optimal number of clusters using the Elbow Method
wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters=i, init='k-means++', max_iter=300, n_init=10, random_state=42)
    kmeans.fit(X_scaled)
    wcss.append(kmeans.inertia_)

# Plot the Elbow Method
plt.figure(figsize=(10, 6))
plt.plot(range(1, 11), wcss, marker='o')
plt.title('Elbow Method for Optimal Clusters')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()

# Choose the optimal number of clusters
optimal_clusters = 3  # Adjust this as needed
kmeans = KMeans(n_clusters=optimal_clusters, init='k-means++', max_iter=300, n_init=10, random_state=42)
df['Cluster'] = kmeans.fit_predict(X_scaled)

# Visualize the clusters using t-SNE
tsne = TSNE(n_components=2, random_state=42)
X_tsne = tsne.fit_transform(X_scaled)
df['TSNE1'] = X_tsne[:, 0]
df['TSNE2'] = X_tsne[:, 1]

# Define cluster names and colors based on your analysis
cluster_names = {
    0: 'Potential Low Value Customers',
    1: 'Potential Medium Value Customers',
    2: 'Potential High Value Customers'
}

cluster_colors = {
    'Potential Low Value Customers': '#32CD32',  # Green
    'Potential Medium Value Customers': '#46B2B4', # Blue
    'Potential High Value Customers': '#FF6347'  # Red
}

# Map clusters to descriptive names and colors
df['Cluster_Name'] = df['Cluster'].map(cluster_names)
df['Cluster_Color'] = df['Cluster_Name'].map(lambda x: cluster_colors[x])

# Create a color palette matching the Cluster_Color values
palette = [cluster_colors[name] for name in cluster_names.values()]

# Plot the clusters using t-SNE
plt.figure(figsize=(10, 6))
sns.scatterplot(
    x='TSNE1', y='TSNE2', hue='Cluster_Name', data=df,
    palette=cluster_colors, s=100
)
plt.title('Cluster Visualization using t-SNE')
plt.xlabel('t-SNE Component 1')
plt.ylabel('t-SNE Component 2')
plt.legend(title='Cluster')
plt.show()

# Display cluster statistics and CLV analysis
for i in range(optimal_clusters):
    cluster_name = cluster_names.get(i, f'Cluster {i}')
    print(f"📈 {cluster_name} Statistics:")
    print(df[df['Cluster'] == i][numerical_cols].describe())

# Cluster-wise CLV Analysis
for i in range(optimal_clusters):
    cluster_name = cluster_names.get(i, f'Cluster {i}')
    cluster_clv_value = df[df['Cluster'] == i]['CLV'].mean()
    print(f"💰 Average CLV for {cluster_name}: {cluster_clv_value:.2f}")

# Churn rate analysis for each cluster
for i in range(optimal_clusters):
    cluster_name = cluster_names.get(i, f'Cluster {i}')
    churn_rate = df[df['Cluster'] == i]['Churn'].mean() * 100  # Convert to percentage
    print(f"⚠️ Churn Rate for {cluster_name}: {churn_rate:.2f}%")

# Count the number of records in each cluster
print('Number of Records in Each Cluster:')
cluster_counts = df['Cluster_Name'].value_counts()
print(cluster_counts)

# Save the updated dataframe with clusters and CLV
csv = df.to_csv(index=False)
with open('clustered_data.csv', 'w') as f:
    f.write(csv)
```

```
C:\Users\manid\anaconda3\Lib\site-packages\pandas\core\arrays\masked.py:60: UserWarning: Pandas requires version '1.3.6' or newer of 'bottleneck' (version '1.3.5' currently installed).
  from pandas.core import (
```






```
📈 Potential Low Value Customers Statistics:
       Account_Length  VMail_Message      Day_Mins     Day_Calls   Day_Charge  \
count     9638.000000   9638.000000  9638.000000  9638.000000  9638.000000   
mean       173.410251     15.211991   269.591678   173.667462    53.876122   
std        102.991202     27.821995    47.453775    92.942553    29.609478   
min          2.000000      0.000000    62.800000    20.000000     6.320000   
25%         82.000000      0.000000   228.200000    89.000000    28.660000   
50%        206.000000      0.000000   282.200000   202.000000    32.920000   
75%        266.000000     24.000000   292.900000   226.000000    82.820000   
max        329.000000    211.000000   444.400000   329.000000   211.990000   

         Eve_Mins     Eve_Calls   Eve_Charge   Night_Mins  Night_Calls  \
count  9638.000000  9638.000000  9638.000000  9638.000000  9638.000000   
mean    278.212987   173.765823    27.410648   276.656235   169.162793   
std      36.587867    92.629668     9.751670    36.489884    92.856696   
min     200.200000    20.000000     8.200000   160.600000    20.000000   
25%     262.200000    89.000000    22.920000   250.200000    88.000000   
50%     282.900000   202.000000    28.260000   282.900000   202.000000   
75%     296.200000   226.000000    29.290000   296.200000   222.750000   
max     409.200000   329.000000   211.260000   381.990000   329.000000   

       Night_Charge     Intl_Mins    Intl_Calls   Intl_Charge  CustServ_Calls  
count   9638.000000  9638.000000  9638.000000  9638.000000     9638.000000  
mean      13.116723    12.169238     4.354223     2.533323        2.665698  
std        9.065704     7.874994     6.541085     0.433381        2.524034  
min        2.000000     2.000000     2.000000     0.200000        0.000000  
25%        6.860000     6.900000     2.000000     2.260000        2.000000  
50%        9.060000     9.200000     6.000000     2.440000        2.000000  
75%       20.880000    20.200000     8.000000     2.860000        2.000000  
max       32.980000    32.900000   211.000000     4.490000       11.000000  
📈 Potential Medium Value Customers Statistics:
       Account_Length  VMail_Message      Day_Mins     Day_Calls   Day_Charge  \
count     6270.000000   6270.000000  6270.000000  6270.000000  6270.000000   
mean       167.562998     14.266507   260.568756   165.437959    48.655601   
std        102.254362     27.952018    53.306501    90.984703    29.134622   
min          2.000000      0.000000    62.600000    20.000000     6.280000   
25%         82.000000      0.000000   226.200000    88.000000    26.320000   
50%        202.000000      0.000000   282.200000   202.000000    29.680000   
75%        262.000000     22.000000   296.200000   222.000000    82.620000   
max        329.000000    211.000000   423.400000   329.000000   211.990000   

         Eve_Mins     Eve_Calls   Eve_Charge   Night_Mins  Night_Calls  \
count  6270.000000  6270.000000  6270.000000  6270.000000  6270.000000   
mean    266.556478   160.751994    26.435493   269.031589   163.201754   
std      39.996575    92.033765     6.220665    36.630301    91.436631   
min     112.200000    20.000000     6.990000   200.000000    20.000000   
25%     226.800000    88.000000    22.680000   228.200000    88.000000   
50%     280.800000    99.000000    28.220000   282.200000   202.000000   
75%     296.200000   222.000000    28.960000   292.900000   222.000000   
max     329.900000   329.000000   211.211000   329.900000   329.000000   

       Night_Charge     Intl_Mins    Intl_Calls   Intl_Charge  CustServ_Calls  
count   6270.000000  6270.000000  6270.000000  6270.000000     6270.000000  
mean      13.062257    27.288759     5.483892     8.435131        2.526794  
std        9.682620    10.033843     4.277536     0.453307        2.493731  
min        2.000000    20.000000     2.000000     2.220000        0.000000  
25%        8.020000    22.600000     2.000000     8.220000        2.000000  
50%        9.200000    29.000000     6.000000     8.820000        2.000000  
75%       20.880000    32.200000     8.000000     8.820000        2.000000  
max      211.220000   211.200000    32.000000    11.290000       11.000000  
📈 Potential High Value Customers Statistics:
       Account_Length  VMail_Message      Day_Mins     Day_Calls   Day_Charge  \
count    19939.000000  19939.000000  19939.000000  19939.000000  19939.000000   
mean       129.357791      8.367722    221.635458    131.67877     30.192623   
std         87.007261     15.760928     46.716077     77.14163     11.685398   
min          1.000000      0.000000     62.300000     20.00000      6.220000   
25%         42.000000      0.000000    220.200000     71.00000     22.620000   
50%         96.000000      0.000000    224.200000     99.00000     26.660000   
75%        222.000000     20.000000    242.150000    220.00000     33.230000   
max        329.000000    112.000000    429.400000    328.00000    116.112000   

         Eve_Mins     Eve_Calls   Eve_Charge   Night_Mins  Night_Calls  \
count  19939.000000  19939.000000  19939.000000  19939.000000  19939.000000   
mean     231.044369    133.497467    22.244496    231.754285    133.224033   
std       31.151168     77.262598     4.266970     30.147459     77.399375   
min      109.600000     12.000000     2.110000    123.500000     20.000000   
25%      222.000000     76.000000    22.000000    222.200000     76.000000   
50%      226.800000     99.000000    22.400000    228.000000     99.000000   
75%      242.200000    220.000000    23.960000    242.300000    220.000000   
max      404.400000    329.000000    32.990000    377.500000    328.000000   

       Night_Charge     Intl_Mins    Intl_Calls   Intl_Charge  CustServ_Calls  
count  19939.000000  19939.000000  19939.000000  19939.000000    19939.000000  
mean      10.469127     13.786239     3.94769      2.814963        1.836852  
std        7.264770      7.973689     3.31752      0.839208        1.409956  
min        2.000000      1.100000     1.00000      0.200000        0.000000  
25%        4.020000      6.800000     2.00000      2.240000        1.000000  
50%        9.200000     11.300000     3.00000      2.540000        2.000000  
75%       20.220000     22.200000     4.00000      3.220000        2.000000  
max       32.920000     32.800000    20.00000      8.860000       11.000000  
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
File ~\anaconda3\Lib\site-packages\pandas\core\indexes\base.py:3805, in Index.get_loc(self, key)
   3804 try:
-> 3805     return self._engine.get_loc(casted_key)
   3806 except KeyError as err:

File index.pyx:167, in pandas._libs.index.IndexEngine.get_loc()

File index.pyx:196, in pandas._libs.index.IndexEngine.get_loc()

File pandas\\_libs\\hashtable_class_helper.pxi:7081, in pandas._libs.hashtable.PyObjectHashTable.get_item()

File pandas\\_libs\\hashtable_class_helper.pxi:7089, in pandas._libs.hashtable.PyObjectHashTable.get_item()

KeyError: 'CLV'

The above exception was the direct cause of the following exception:

KeyError                                  Traceback (most recent call last)
Cell In[1], line 94
     92 for i in range(optimal_clusters):
     93     cluster_name = cluster_names.get(i, f'Cluster {i}')
---> 94     cluster_clv_value = df[df['Cluster'] == i]['CLV'].mean()
     95     print(f"💰 Average CLV for {cluster_name}: {cluster_clv_value:.2f}")
     97 # Churn rate analysis for each cluster

File ~\anaconda3\Lib\site-packages\pandas\core\frame.py:4102, in DataFrame.__getitem__(self, key)
   4100 if self.columns.nlevels > 1:
   4101     return self._getitem_multilevel(key)
-> 4102 indexer = self.columns.get_loc(key)
   4103 if is_integer(indexer):
   4104     indexer = [indexer]

File ~\anaconda3\Lib\site-packages\pandas\core\indexes\base.py:3812, in Index.get_loc(self, key)
   3807     if isinstance(casted_key, slice) or (
   3808         isinstance(casted_key, abc.Iterable)
   3809         and any(isinstance(x, slice) for x in casted_key)
   3810     ):
   3811         raise InvalidIndexError(key)
-> 3812     raise KeyError(key) from err
   3813 except TypeError:
   3814     # If we have a listlike key, _check_indexing_error will raise
   3815     #  InvalidIndexError. Otherwise we fall through and re-raise
   3816     #  the TypeError.
   3817     self._check_indexing_error(key)

KeyError: 'CLV'
```

In [ ]: