D212 - Data Mining II - Task 1

Jeremy Dorrough

Western Governors University

WGU Student ID# 000994113

A1.

Can k-means clustering be used to identify customer segments based on their tenure and supporting demographic and behavior features?

A2.

One goal of this data analysis is to discover customer segments that have similar characteristics by using tenure and supporting features.

B1.

K-means clustering was selected because it can be used to analyze the dataset and identify distinct customer segments based on similarity within the selected features. The expected outcome is that appropriate customer segments will be found and can be used to identify the factors that are contributing to customer tenure.

B2.

An assumption of k-means clustering technique is that clusters are relatively similar in size and do not have a significant deviation in their size. It assumes the data is isotropic, with an equal variance across axes.

B3.

The following Python packages and libraries were used in the analysis. Additionally, magic commands are noted:

|  |  |
| --- | --- |
| Packages/Libraries:  *Note: includes magic commands.* | Usage Justification: |
| pandas | Data manipulation, working with data frames, summary statistics. |
| scipy.stats | Statistical calculations & outlier removal. |
| numpy | Work with arrays and matrices. |
| matplotlib.pyplot | Create histograms, scatterplots for data visualization. |
| seaborn | Create visualizations, particularly scatterplots. |
| matplotlib.lines | Create custom legend elements for data visualizations. |
| %matplotlib inline | Display plots inline in Jupyter Notebooks. |
| %config InlineBackend... | Increase plot resolution in Jupyter Notebooks. |
| sklearn.cluster | Perform k-means clustering, group customers based on characteristics. |

C1.

One goal of the data preprocessing was to standardize continuous variables so that they were all on the same scale before performing k-means clustering. This insured that the information was appropriately distributed and correctly represented in the clustering process.

C2.

The following dataset variables contained information relevant to the business question and were used for the analysis. They are all continuous variables as labeled below:

|  |  |  |
| --- | --- | --- |
| 1. | Tenure | Continuous |
| 2. | Income | Continuous |
| 3. | Age | Continuous |
| 4. | Children | Continuous |
| 5. | MonthlyCharge | Continuous |
| 6. | Bandwidth\_GB\_Year | Continuous |

C3.

1. Imported and loaded the Packages and Data.

The packages were imported, which included pandas, scipy.stats, seaborn, matplotlib.pyplot, numpy, sklearn.cluster, sklearn.preprocessing, and warnings. The churn data was loaded from the provided CSV file into a pandas data frame.

# Data manipulation

import pandas as pd

# Calculations, removing outliers

import scipy.stats as stats

import numpy as np

# Clustering and Scaling

from sklearn.cluster import KMeans

from sklearn.preprocessing import StandardScaler

# Silhouette Score

from sklearn.metrics import silhouette\_score

# Data visualization

import matplotlib.pyplot as plt

import seaborn as sns

from matplotlib.lines import Line2D

# Increase resolution, display plots inline

%config InlineBackend.figure\_format = 'retina'

%matplotlib inline

#Import Data

df = pd.read\_csv('~/Desktop/D212/churn\_clean.csv')

#Examine shape

df.shape

2. Selected the initial variables used in the analysis.

The data was reduced so that the dataframe included only the initial variables for the analysis (Tenure, Income, Age, Children, MonthlyCharge, and Bandwidth\_GB\_Year).

#Select initial variables

df = df[['Tenure', 'Income', 'Age', 'Children', 'MonthlyCharge', 'Bandwidth\_GB\_Year']]

#Examine reduced dataset

print(df)

3. Detected and Removed Duplicates and Missing Values

The data frame was checked to see if there were duplicate rows, none were detected. Additionally, it was checked for missing values, and none were detected.

#Check for Duplicates

df.duplicated().sum()

#Check for Missing Values

df.isnull().sum()

4. Detected and Removed Outliers

Outliers were detected and removed. This was done by calculating the z-scores and storing them in respective columns for each variable. Outlier rows were dropped based on a standard deviation value (as indicated in the z-score columns) of three or less than negative three. The z-score columns were then dropped.

# Create Z-score columns for selected features

zscore\_cols = ['Children', 'Age', 'Income', 'Tenure', 'MonthlyCharge', 'Bandwidth\_GB\_Year']

for col in zscore\_cols:

df[f"{col}\_Zscore"] = stats.zscore(df[col])

# Drop rows with z-scores higher than 3 or lower than -3 for selected features

for col in zscore\_cols:

df.drop(df[(df[f"{col}\_Zscore"] > 3) | (df[f"{col}\_Zscore"] < -3)].index, inplace=True)

# Drop z-score columns

df.drop([f"{col}\_Zscore" for col in zscore\_cols], axis=1, inplace=True)

df.shape

5. Examined the Results

The dataset was checked following the cleaning which included examining individual values by printing the header and identifying how many rows remained after cleaning by examining the data frame’s shape. The summary statistics were examined for all variables and both univariate and bivariate visualizations were generated to explore data distribution and the variable relationships.

#Examine results  
df.info()

df.head()

#View Summary Statistics

df.describe()

#Univariate Visualizations

fig = df.hist(figsize=(12, 10))

[x.title.set\_size(10) for x in fig.ravel()]

plt.tight\_layout()

plt.show()

#Bivariate visualizations

# Get features except Tenure/last column

features = [col for col in df.columns if col not in ['Tenure']]

# Set rows/columns

n\_rows = 3

n\_cols = 2

# Figure and subplot array

fig, axes = plt.subplots(n\_rows, n\_cols, figsize=(12, 10))

# Scatterplots

for i, feature in enumerate(features):

row = i // n\_cols

col = i % n\_cols

sns.scatterplot(x='Tenure', y=feature, data=df, ax=axes[row, col])

# Fix spacing

fig.tight\_layout()

# Show plot

plt.show()

6. Scaled the Data

Prior to performing the k-means clustering, the data was standardized into a scale from -1 to 1 to ensure that all features were correctly and equally weighted for clustering, as clustering using their original scales could distort the results of the analysis.

# Scale the data

scaler = StandardScaler()

data\_scaled = scaler.fit\_transform(df)

7. Exported Preprocessed Data

The final step of the preprocessing was to export the prepared dataset to a CSV file.

# Export to CSV

df.to\_csv("~/Desktop/D212/preprocessed\_data.csv")

C4.

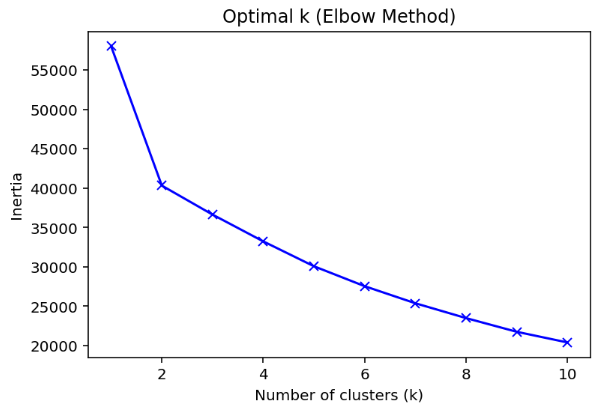
Please find the cleaned dataset attached as “preprocessed\_data.csv”.

D1.

K-means clustering was the analysis technique used for this analysis. It’s an unsupervised machine learning algorithm that groups the selected data into clusters based on patterns it detects (Klingensmith, 2023). A K value is determined, which represents the number of clusters. In this case the elbow method was used to determine the value. Following this step, the K-means technique selects a random K centroid for the clusters, assigns the data points to the nearest centroid, generally using Euclidean distance. It then determines the mean value for every cluster. This process is repeated by moving the centroid until the best location is identified or until it reaches a preset max number of iterations. This process identifies the final clusters for the data and their centroid.

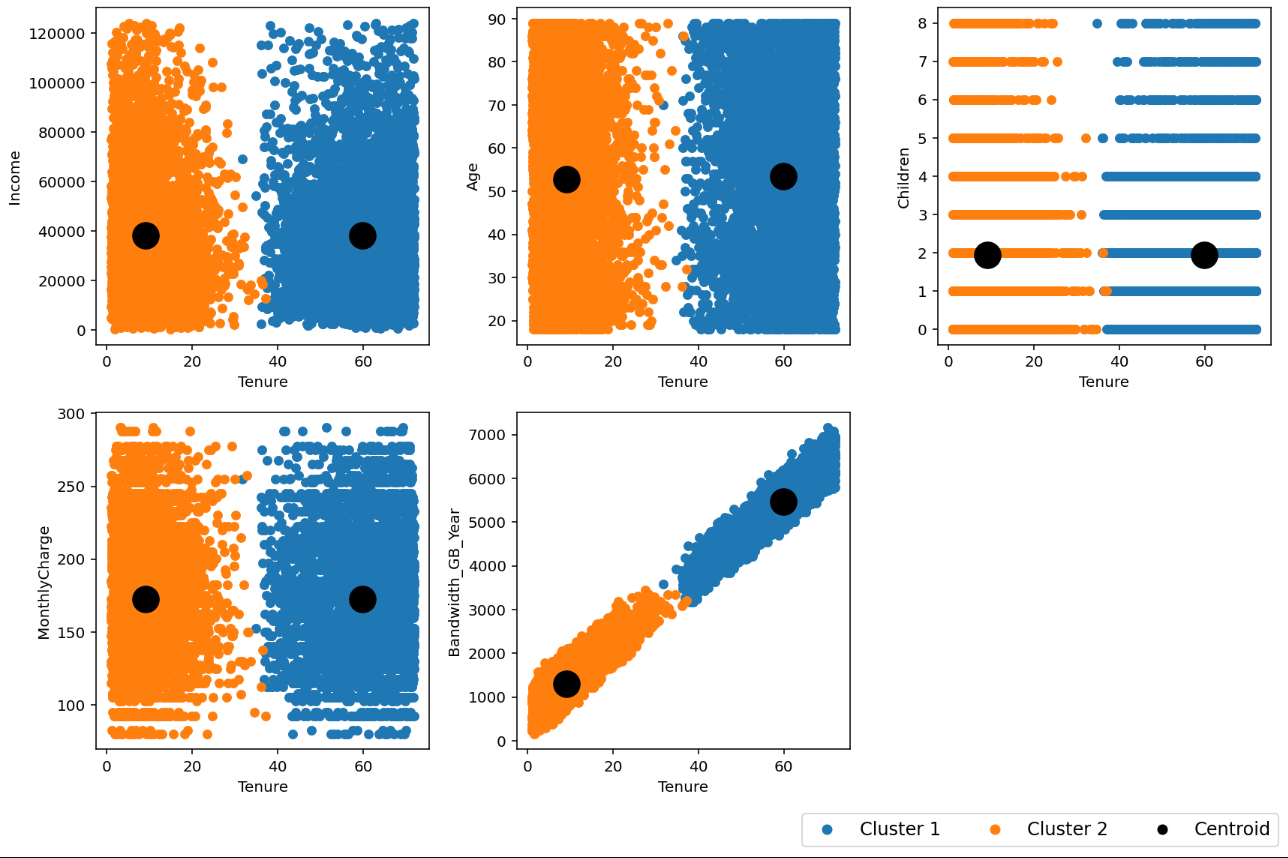
Here, K-means clustering was used to find customer clusters using tenure along with the supporting features of income, age, monthly charge, bandwidth usage, and number of children. Note that the data was prepared, including being scaled, as described in section C3 above. An intermediate calculation was performed to identify the optimal-k value of 2 using the Elbow Method and the data was clustered using that value, as can be examined in Figure 1 below.

Figure 1:



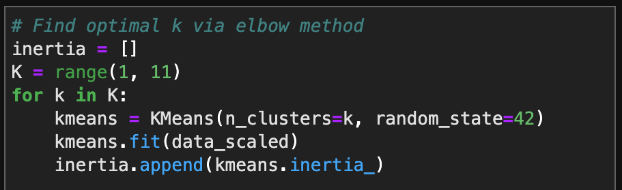
The results were then visualized, including the centroid on a grid of scatter plots for each variable’s relationship with tenure, which was generated by looping through the clusters for each feature, plotting the data into color coded clusters with a prominent black dot for the centroids, which can be seen in Figure 2 below. Finally, a silhouette score was calculated to help determine how accurate the clustering was and the calculation screenshot can be examined in.

Figure 2:

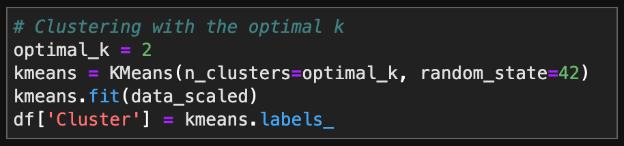


Screenshots of intermediate calculations performed within Python can be examined below. Additionally, full code can be examined in section D2:

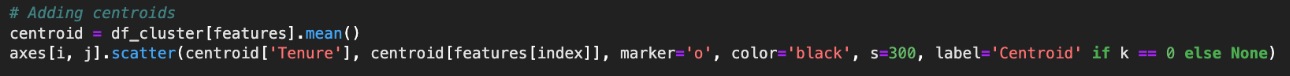
Calculation to discover optimal k using the elbow method:



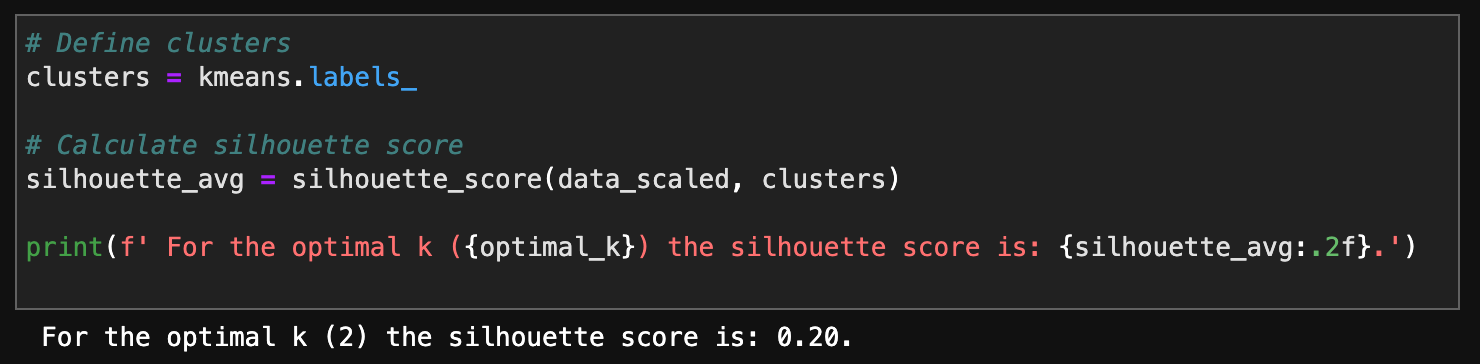
Calculation to define optimal k and use it for clustering:



Calculation to identify centroids. Note: also used to plot centroids, see D4 for complete code:



Calculation to define clusters (used to determine silhouette score):



D2.

The below code performs the k-means clustering and creates scatterplots to visualize the clusters, including the centroid, for the relationships between tenure and the other features. It also calculates and prints the silhouette score.

# Clustering with the optimal k (Arvai, n.d.)

optimal\_k = 2

kmeans = KMeans(n\_clusters=optimal\_k, random\_state=42)

kmeans.fit(data\_scaled)

df['Cluster'] = kmeans.labels\_

# Get columns

features = [col for col in df.columns[:-1] if col != 'Tenure']

# Set Rows/Columns

n\_rows = 2

n\_cols = 3

# Set subplot size

subplot\_width = 4

subplot\_height = 4

# Set figure size

fig\_width = n\_cols \* subplot\_width

fig\_height = n\_rows \* subplot\_height

# Create figure/subplots

fig, axes = plt.subplots(n\_rows, n\_cols, figsize=(fig\_width, fig\_height))

# Add legend labels

clusters = ['Cluster 1', 'Cluster 2']

# Create scatterplots for features vs tenure (Venmani A D., n.d.; PYnative, 2021)

for i in range(n\_rows):

for j in range(n\_cols):

index = i \* n\_cols + j

if index < len(features):

for k in range(len(df['Cluster'].unique())):

df\_cluster = df[df['Cluster'] == k]

axes[i, j].scatter(df\_cluster['Tenure'], df\_cluster[features[index]], s=30, label=clusters[k])

# Adding centroids

centroid = df\_cluster[features].mean()

axes[i, j].scatter(df\_cluster['Tenure'].mean(), centroid[features[index]], marker='o', color='black', s=300, label='Centroid' if k == 0 else None)

# Add x and y labels

axes[i, j].set\_xlabel('Tenure')

axes[i, j].set\_ylabel(features[index])

# Drop empty subplots

for i in range(len(features), n\_rows \* n\_cols):

fig.delaxes(axes.flatten()[i])

# Fix spacing

fig.tight\_layout(rect=[0, 0.05, 1, 1])

# Create legend

legend\_elements = [Line2D([0], [0], marker='o', color='w', label=label, markerfacecolor='C'+str(i), markersize=8) for i, label in enumerate(clusters)]

legend\_elements.append(Line2D([0], [0], marker='o', color='w', label='Centroid', markerfacecolor='black', markersize=8))

# Add legend

fig.legend(handles=legend\_elements, loc='lower right', ncol=len(clusters) + 1, bbox\_to\_anchor=(1, 0), fontsize='large')

# Show plots

plt.show()

# Define clusters

clusters = kmeans.labels\_

# Calculate silhouette score

silhouette\_avg = silhouette\_score(data\_scaled, clusters)

print(f' For the optimal k ({optimal\_k}) the silhouette score is: {silhouette\_avg:.2f}.')

E1.

The accuracy of the clustering technique was evaluated by using a silhouette score, which was determined to be 0.30. In silhouette scoring, scores range from -1 to 1 and higher values indicate better clustering (scikit-learn.org). Since the score was 0.30, the clustering had a moderate separation. It’s important to note that the clear bimodal distribution across all relationships with the tenure variable with a gap in the tenures at approximately the 25-40 months mark. This indicates that there is a meaningful structure in the data and in the clusters, which correctly separated those into two distinct groups.

E2.

The analysis discovered two distinct customer groups based on the provided characteristics, and the centroids were centrally located within each cluster. As previously stated, it illustrated a prominent gap in tenure at approximately the 25-40 months mark. An implication of these results is that there are different customer segments, as it relates to tenure, that may have different behaviors, needs, and preferences which aren’t adequately addressed or understood. It demonstrates a problem facing the company, as it relates to maintaining tenure, in the approximated 25-40 period.

E3.

A limitation of the analysis was that it only analyzed a limited number of features from the available dataset. Other features may have provided more insights and made the analysis more accurate. Considering a silhouette score of only 0.30, the resulting clustering may have achieved a higher score, and ultimately have been more accurate for real-world business decision-making, if other potentially relevant variables had been included.

E4.

The organization should utilize the analysis results to inform business decision making. The observations can be used to create better strategies to engage customers such as new pricing options, marketing techniques, or retention programs that are more specifically designed to engage each segment. Investigating, understanding, and addressing the gap in tenure may present a different set of strategies for customer engagement. This, for example, could ultimately result in adjusting pricing, bandwidth offerings, or other contract options. Additionally, this customer segmentation should continue to be refined and used for ongoing evaluation.

F.

Please find Panopto presentation attached: <https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=525defb6-3fd6-41e0-883c-afce0177b99e>

G.

Arvai, K. (n.d.). K-Means Clustering in Python: A Practical Guide. Real Python. Retrieved from <https://realpython.com/k-means-clustering-python/>

Venmani A D. (n.d.). Python Scatter Plot – Complete Guide. Machine Learning Plus. Retrieved from <https://www.machinelearningplus.com/plots/python-scatter-plot/>

PYnative. (2021). Python Nested Loops [With Examples]. Retrieved from <https://pynative.com/python-nested-loops/>

H.

Klingensmith, K. (2023). K-means Clustering: An Introductory Guide and Practical Application. Towards Data Science. Retrieved from <https://towardsdatascience.com/k-means-clustering-an-introductory-guide-and-practical-application-dce70bfa4249>

scikit-learn.org. (n.d.). sklearn.metrics.silhouette\_score — scikit-learn 1.2.2 documentation. Retrieved March 23, 2023 from <https://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html>