**K-means Clustering of Customers Based on Tenure and Monthly Charge**

Kelseyann Wright

Student ID: 00565814

College of Information Technology, Western Governors University

Data Mining II - D212

Kesselly Kamara

December 9, 2023

**K-means Clustering of Customers Based on Tenure and Monthly Charge**

**A1. PROPOSAL OF QUESTION**

The proposed question is, “Can customers be grouped based on tenure and monthly charge?”. This question is important because it could potentially identify patterns in our customer base that may aid in marketing and customer retention.

**A2. DEFINED GOAL**

The goal of this analysis is to use k-means clustering to group customers based on tenure and monthly charge to identify patterns in the customer population that may aid in marketing and customer retention.

**B1. EXPLANATION OF CLUSTERING TECHNIQUE**

K-means clustering is an unsupervised machine learning algorithm which assigns each data point to one of k “clusters”, or groups. The centroid of each cluster is calculated as the mean of the observations in that cluster, and the distortion, or the sum of the squared distances between the cluster points and the centroid are used to determine the model’s clustering success. It is an iterative process, meaning each point is assigned to a cluster and the centroids and distortions are re-calculated and adjusted until the algorithm converges. Convergence is achieved when the algorithm achieves its goal of minimizing the distortion within a group while maximizing the distance between groups.

**B2. SUMMARY OF THE TECHNIQUE ASSUPTION**

One assumption of k-means clustering is that each cluster has approximately the same number of observations. This means that the algorithm may struggle to identify unique clusters if they are not all relatively equal in size (i.e., number of data points). Smaller clusters that are far away from the centroid have less of an impact on the cluster sum of squared distances, which the algorithm attempts to minimize. In datasets with small and large clusters, the k-means algorithm may split up larger clusters, while assigning the smaller cluster(s) to the closest of centroid of those larger groups. This will minimize the cluster sum of squared distances but will not identify truly unique clusters.

**B3. PACKAGES OR LIBRARIES LIST**

The libraries used in the analysis are described in Table 1.

***Table 1: Justification of python libraries used in the analysis.***

|  |  |  |
| --- | --- | --- |
| Library | Justification / Use | Source |
| Pandas | Data frame manipulation and summarization | McKinney, 2010 |
| numpy | Data manipulation and formatting for statistical analyses | Harris et al., 2020 |
| matplotlib | Data visualization | Hunter, 2007 |
| Sci-kit learn | Includes a function for splitting data into train and test sets and a function for executing the Naïve Bayes algorithm. This library also includes multiple functions for testing model accuracy. | Pedregosa et al., 2011 |
| kneed | Includes a function to identify the “knee” point of a function used to identify the number of clusters for the k-means algorithm. | Satopaa et al., 2011 |

**C1. DATA PROCESSING**

One data processing goal relevant to the k-means algorithm is feature scaling. The data were standardized using the StandardScaler() function from the sci-kit learn library in python (Pedregosa et. al 2011). This function scales features so that they will have a Gaussian normal distribution with a mean of zero and a standard deviation of 1. Standardizing data is important for k-means clustering because it helps the algorithm apply similar weight to each feature when minimizing the sum of squared distances for data clusters.

**C2. DATA SET VARIABLES**

Two variables were used for this analysis: tenure and monthly charge. Both variables are continuous numeric variables, which is required for the k-means algorithm.

**C3. STEPS FOR ANALYSIS**

First, the two variables to be used for k-means clustering were isolated from the dataset and saved to a new dataframe.

df = dat[["Tenure", "MonthlyCharge"]]

Second, the data were checked for duplicates and nulls. None were discovered.

print("number of duplicates:", df.duplicated().sum())

print("number of nulls:", df.isna().sum().sum())

Finally, the data were standardized as described in section C1.

#scale features using standard scalar

scaler = StandardScaler()

df\_mod = pd.DataFrame(scaler.fit\_transform(df), columns = df.columns)

**C4. CLEANED DATA SET**

Two CSV files containing cleaned datasets are attached: 1) the cleaned dataset prior to standardization (churn\_processed\_T1.csv), and after standardization (churn\_scaled\_T1.csv).

**D1. OUTPUT AND INTERMEDIATE CALCULATIONS**

The submission determines the optimal number of clusters in the data set and accurately describes the methodology used. The methodology is appropriately applied.

A graph with a blue line

Description automatically generated To identify the optimal number of clusters, the k-means algorithm was run in a loop with various numbers of clusters, ranging from 1-10. The resulting inertia for each algorithm was plotted against the number of clusters used (Figure 1). Because it was difficult and subjective to identify the optimal number of clusters from the curve of the inertia plot (Figure 1), the kneedle algorithm (Satopaa et al., 2011) was used to identify the knee of the curve. The kneedle algorithm was performed using the “KneeLocator” function in the “kneed” python package. This package identified the elbow of the curve, and therefore the optimum number of clusters to use, was 4 (see output below).

Figure : Comparison of k-means algorithm inertias for different numbers of clusters.

A black screen with white text

Description automatically generated

**D2. CODE EXECUTION**

The code used for all data preparation and analysis is attached (D212\_T1.py).

**E1. QUALITY OF THE CLUSTERING TECHNIQUE**

The k-means algorithm identified four clusters in the dataset using the provided features of tenure and monthly charge (Figure 2, Figure 3).

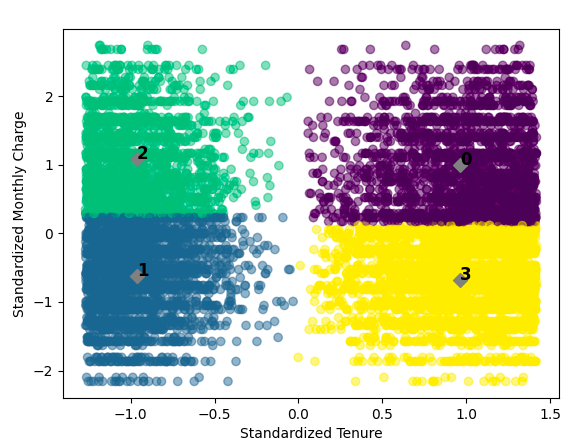


Figure 2: Scatterplot of clusters identified based on tenure and monthly charge with labelled centroids for each cluster. Monthly charge and tenure are standardized.

Figure : Barplot of mean tenure and monthly charge for identified clusters

**A graph of different colored bars

Description automatically generated**

A graph of a number of blue and white bars

Description automatically generatedA graph of a graph

Description automatically generatedCluster “0” has the largest tenure and monthly charge values (Figure 2, Figure 3). Cluster “1” has the lowest tenure and monthly charge values (Figure 2, Figure 3). Cluster “2” has higher monthly charge values but and lower tenure (Figure 2, Figure 3). Cluster “3” has higher tenure and lower monthly charge (Figure 2, Figure 3). Although each cluster has unique combinations of high/low tenure and monthly charge (Figure 3), the scatterplot shows that there is not clear separation of observations between clusters 1 and 2 as well as clusters 0 and 3 along the “monthly charge” axis (Figure 2). Based on Figure 2, it appears there is a clear separation of clusters based on tenure, but the separation based on monthly charge is less distinct. This is supported by the histograms of the two features (below).

In the histograms (above), you can see that there are two distinct groups of observations based on “Tenure” (left above), but not “Monthly Charge” (right above).

Using the original, not scaled data in the scatterplot (Figure 4)

**E2. RESULTS AND IMPLICATIONS**

The results indicate that customers can be clustered into four groups based on tenure and monthly charge, however the separation based on monthly charge is not as distinct as the separation based on tenure. The four clusters of customers identified were:

Group 0: high tenure, high monthly charge

Group 1: low tenure, low monthly charge

Group 2: low tenure, high monthly charge

Group 3: high tenure, low monthly charge

The separation based on tenure is obvious even when viewing a histogram of the customer data. This clustering analysis implies that tenure may be a good feature for grouping customers, but monthly charge may be unrelated to this grouping.

**E3. LIMITATION**

One limitation of the k-means cluster algorithm is that it is highly sensitive to outliers in the data. Because the cost function of the algorithm is the sum of squared distances, outliers can drastically affect the reliability of the clusters. This is compounded by the assumption of the algorithm that all clusters are the same size, meaning that a small number of outliers will not be classified as their own cluster(s) but rather be forced into other clusters, again decreasing the reliability of the clusters.

**E4. COURSE OF ACTION**

This analysis indicates that there are four clusters of customers based on tenure and monthly charge. However, there is little separation between customers along monthly charge axis, and the “cut-off” between low and high monthly charge appears arbitrary. My recommended course of action would be to explore other features with customer tenure to improve clusters. In addition, it may be necessary to try other clustering techniques that do not assume equal sized clusters to represent the customers more accurately and produce more reliable clusters.

**F. PANOPTO VIDEO OF CODE AND F1. PANOPTO VIDEO OF PROGRAMS**

A Panopto video providing full documentation of the code and description of programs used is attached. The link is also provided here: <https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=f8923b74-5128-4496-a502-b0ea01189ae9>

**G. SOURCES OF THIRD-PARTY CODE**

No sources of third-party code were used.

**H. SOURCES**

Harris, C.R., Millman, J.K., van der Walt, S.J, Gommers, R., Virtanen, P., Cournapeau, V., Wieser, E., Taylor, J., Berg, S., Smith, N.J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M.H., Brett, M., Haldane, A., Fernandez del Rio, F., Wiebe, M., Peterson, P., Gerard-Marchant, P., Sheppard, K., … Oliphant, T.E. (2020). Array programming with NumPy. Nature, 585(7825), 357-362.

Hunter, J.D. (2007). "Matplotlib: A 2D Graphics Environment", Computing in Science & Engineering, vol. 9, no. 3, pp. 90-95.

McKinney, W. (2010). Data structures for statistical computing in python. In Proceedings of the 9th Python in Science Conference (Vol. 445, pp. 51–56).

Pedregosa, F., Varoquaux, Ga"el, Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., and Dubourg, V. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12(Oct), 2825–2830.

Satopaa, V., Albrecht, J., Irwin, D., & Raghavan, B. (2011, June). Finding a" kneedle" in a haystack: Detecting knee points in system behavior. In *2011 31st international conference on distributed computing systems workshops* (pp. 166-171). IEEE.