Using k-means, can we effectively group customers with similar characteristics to create customer segmentation for company use?

By:

Kevin Sandoval

**A1.**

I attempted to re-clone the D603 Machine Learning repository, but since there isn’t a separate one from Task 1 and 2, it did not work. Thus I am using the same repo from Task 1 but creating additional branches. The branches in Task 2 will be in the format “Task2\_Part#\_Number\_#\_Update#” and Task 1 will be in the format of “Task1\_Part#\_Number\_#\_Update#”.

**A2.**

Each part of D and E have their own committed branch on the repo.

**A3.**

A link to the GitLab repository is provided in the “Comments to Evaluator” section.

**A4.**

A .txt file containing the history of the activity from the repo is provided in the submission.

**B1.**

The research questions I will be examining is: “Using k-means, can we effectively group customers with similar characteristics to create customer segmentation for company use?” This question has important implications for a business as retaining customers is a vital part of the company’s success. Examining the variables that can be used to group similar customers might help determine if a new customer is likely to have a long tenure with the company. This could either aide in retaining the new customer or help the company focus more on a different set of similar characteristics for either retention or acquisition.

**B2.**

The goal(s) of the analysis will be to create groups of customers that share similar characteristics by maximizing centroid distance between different clusters while minimizing Euclidean distance between data points within each unique cluster. Another goal of this analysis will be to find the optimal number of clusters for the KMeans technique and then be able to look at the clusters and discuss the quality of them. I will then use these clusters to recommend a course of action for the company.

**C1.**

KMeans is the clustering technique that I chose to analyze the set. This works in analyzing the data set because I am looking at the continuous variables to try and create an effective cluster for customer segmentation. The clustering technique will then create groups of customers based on similar characteristics. The expected outcome for this analysis will be to maximize the distance between each unique cluster’s centroid. The further the distance between each cluster’s centroid creates better results and easier to interpret clusters. Another expected outcome will be to minimize the Euclidean distance between data points in each unique cluster. What this means is that for each cluster, there should not be any points that are too far away from the given centroid.

The first part of KMeans is the initialization. In this analysis, I used ‘k-means++’ which “selects initial cluster centroids using sampling based on an empirical probability distribution of the points’ contribution to the overall inertia. This technique speeds up convergence.” (scikit-learn, *KMeans*) KMeans then computes the Euclidean distance of each data point to all centroids. Since the objective is to minimize Euclidean distance, each data point gets assigned to the nearest centroid. The next step is to update the centroids by taking the mean of every data point in each respective cluster. KMeans then repeats the process of computing the Euclidean distance for each data point, assigning to the nearest new centroid, and then updating the centroids until it stops and gives its final labels. This is done by checking for convergence; which happens when the centroids do not change significantly between runs, the data points no longer get reclassified and stay as previously assigned, or if the model reaches a set limit of iterations by the user.

**C2.**

One assumption of K-Means clustering is that the clusters are spherical and isotropic. This implies that the clusters’ radius is approximately equal in all directions. Since the cluster center is assigned to the mean that the algorithm finds from averaging the data points of a cluster, K-Means can be susceptible to non-spherical or elliptical clusters. (GeeksforGeeks, *Demonstration of k-means assumptions)* In conjunction with this, another assumption is the number of clusters. KMeans does not find the optimal number of clusters automatically, so it is on the user to use other methods to find the optimal amount. The optimal amount “has to be decided from data-based criteria and knowledge of the intended goal.” (scikit-learn, *Demonstration of k-means assumptions*)

**C3.**

The Python packages I used are explained and shown below.

**Numpy:** Helps preform numerical calculations on arrays

**Pandas:** Allows me to import and export csv files

**Matplotlib.pyplot:** Helpful for visualizing distributions and plots

**Matplotlib.cm:** Helpful for visualizing the silhouette graphs and scores

**Missingno:** Helpful for checking if there are missing values in the dataset

**Seaborn:** Helpful for visualizations

The following are imported from sklearn.metrics

**Silhouette\_samples:** Helps with getting the score for the samples

**Silhouette\_score:** Gives the average value for all the samples

The following are imported from sklearn.cluster

**KMeans:** Clustering technique for the analysis

The following are imported from sklearn.decomposition

**PCA:** Helps with dimensionality reduction

**D1.**

The first thing that I did when I pulled in the data set was to check the shape of the data frame. The shape was (10000 , 50) so I knew that there were 10,000 rows of data with 50 columns. I then checked if there were any duplicate values and got that there were no duplicates.

After that I checked if there were any missing values using the .isna().sum() function to determine if there were any missing values. I found that the data set did not contain any missing values so none were treated. I then decided to check every quantitative variable for outliers. I did find outliers for the columns “Population”, “Children”, “Income”, “Outage\_sec\_perweek”, “Email”, “Contacts”, and “Yearly\_equip\_failure.” I treated the outliers by excluding them to a new variable for each except for “Population” which I decided to retain. After excluding the initial outliers, “Income” and “Outage\_sec\_perweek” still showed that they had a few new outliers, but I decided to retain those as it would be diminishing returns on the exclusion of data. My new data frame shape is (9079, 50).

**D2.**

All variables used in the analysis are listed below with their classification as well.

**Tenure:** Numeric

**Children:** Numeric

**Age:** Numeric

**Income:** Numeric

**Outage\_sec\_perweek:** Numeric

**Email:** Numeric

**Contacts:** Numeric

**Yearly\_equip\_failure:** Numeric

**MonthlyCharge:** Numeric

**Bandwidth\_GB\_Year:** Numeric

**D3.**

**Step 1:** Checking for duplicates

The first step I performed when preparing the data was to check for duplicate values in the data. This is important as duplicate values can have a misleading effect on the outcome of the analysis. The code snippet and output is shown below.

A screenshot of a computer program

Description automatically generated

**Step 2:** Checking for missing values

The 2nd step of preparing the data was to check for missing values in the data. It is important to check as missing values can skew the data and the results. I used the missingno matrix to check for missing values as well as na.sum() to check. The code and some of the output is shown below.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Step 3:** Removing outliers

The third step in treating the data is to remove any outliers. I first checked the boxplot of each quantitative variable and if they showed outliers, I treated them. The code segments for the boxplot and outlier removal is shown below with “Children” as the example variable.

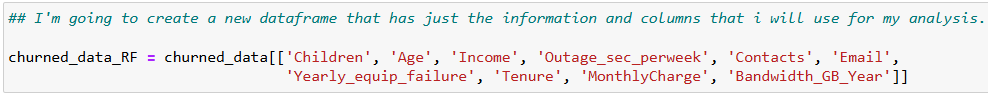


A screen shot of a computer code

Description automatically generated

**Step 4:** Creating new dataframe unneeded columns

In this step I created a new dataframe that only contained the continuous variables I am focused on.



**Step 5**: Performing PCA before KMeans to help with dimensionality reduction as recommended by scikit-learn

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screen shot of a graph

AI-generated content may be incorrect.

From this we can see we would keep PCA1-PCA5.

**D4.**

Prepared CSV is included in the submission.

**E1.**

From my analysis the optimal number of clusters in the dataset appears to be 6. I performed 2 tests to find the optimal number of clusters. I decided to do silhouette plotting and scores per the scikit-learn documentation. After using the code they had provided, the best silhouette score was at 6 clusters. This was a less conclusive results as none of the silhouette scores were particularly great and they were all within 0.02 of each other. Because of this, I decided to do another check for the optimal number of clusters just to be safe. The next test was to create a for loop that checked all clusters from 2 through 9 and then append the inertia of each cluster. After plotting each cluster’s inertia, there is an inflection point on the X-axis at 6, thus implying that the optimal number of clusters would be 6.

**F1.**

The 9 cluster visualizations and silhouette scores are shown below.

A screen shot of a computer screen

AI-generated content may be incorrect.

A screenshot of a graph

AI-generated content may be incorrect.

A screenshot of a graph

AI-generated content may be incorrect.

A screenshot of a graph

AI-generated content may be incorrect.

A screenshot of a graph

AI-generated content may be incorrect.

A screenshot of a graph

AI-generated content may be incorrect.

A screenshot of a graph

AI-generated content may be incorrect.

A screenshot of a graph

AI-generated content may be incorrect.

A screenshot of a graph

AI-generated content may be incorrect.

**F2.**

In this analysis I found that the optimal number of clusters for the dataset was 6 clusters. We can see that in both the silhouette score, as well as the sample inertia graph. When viewing the scatterplot of data, the clearest divide might be the 2 cluster. However, the silhouette score is also the lowest as 2 clusters tend to be the least robust. When viewing the scatterplot for 6 clusters, the most obvious clusters are 1,3, and 4 respectively. Clusters 2,5, and 6 are closer together and more ambiguous to their differences. One implication of this analysis could be that since the data generated is random, it is hard to find patterns in the data. This could lead to muddled results and poor scores in the analysis.

**F3.**

One limitation of the analysis is the variables that are chosen. Because I chose to use KMeans, that requires all variables to be continuous. This obviously reduces the number of variables that can be used in the analysis. There could be important categorical variables that would effectively group customers together, but that was not considered in this analysis.

**F4.**

Although the silhouette scores were not incredible, and the graphs don’t reflect perfect clusters, I do think there is some utility to these visuals. A course of action that I would recommend to the company would be to closely examine the cluster labels “1”, “3”, and “4” from the 6-cluster silhouette analysis. These 3 groups are the clearest as you can visibly see the blue, green, and yellow sections respectively. I would recommend looking into the tenure length of each of those sections. If the tenure length were to increase significantly from cluster “3” to cluster “4”, then the company could focus on trying to find customers that more closely resemble those found in cluster “4”. This would help the company in retention rates as well as finding new customers who are more likely to stay for longer. From this we can see that the K-Means analysis reasonably answers the question posed in B1. We would prefer to see more succinct clusters, but there is still some utility in them despite not being perfect. The company would be able to utilize this information to help their retention rates.

**G:**

Panopto video provided in the submission.

**H/I:**

Dr. Middleton, K. (n.d). *Getting Started with D206 Principal Component Analysis(PCA) .* Retrieved July 8th, 2024,From D206 Course Guide

Geeksforgeeks. (December 9th, 2023). *Demonstration of K-Means Assumption.* Retrieved February 22nd, 2024,From <https://www.geeksforgeeks.org/demonstration-of-k-means-assumptions/>

Scikit-learn. (n.d). *Demonstration of k-means assumptions.* Retrieved February 18th, 2024,From <https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_assumptions.html#sphx-glr-auto-examples-cluster-plot-kmeans-assumptions-py>

Scikit-learn. (n.d). *KMeans.* Retrieved February 17th, 2024,From <https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html>

Scikit-learn. (n.d). *Selecting the number of clusters with silhouette analysis on KMeans clustering.* Retrieved February 15th, 2024,From <https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html#sphx-glr-auto-examples-cluster-plot-kmeans-silhouette-analysis-py>

365DataScience(April 15th, 2024) *How to Combine PCA and K-means Clustering in Python?* Retrieved February 19th, 2025 From: <https://365datascience.com/tutorials/python-tutorials/pca-k-means/>