

# IMPROVING CUSTOMER CHURN PREDICTION THROUGH SEGMENTATION

by

Smit Patel, B.Sc.

University of Ottawa, 2020

A Major Research Paper

presented to Toronto Metropolitan University

in partial fulfillment of the requirements for the degree of

Master of Science

in the program of

Data Science and Analytics

Toronto, Ontario, Canada, 2023

© Smit Patel, 2023

**AUTHOR'S DECLARATION FOR ELECTRONIC SUBMISSION OF A MAJOR RESEARCH  
PAPER (MRP)**

I hereby declare that I am the sole author of this Major Research Paper. This is a true copy of the MRP, including any required final revisions, as accepted by my examiners.

I authorize Toronto Metropolitan University to lend this MRP to other institutions or individuals for the purpose of scholarly research.

I further authorize Toronto Metropolitan University to reproduce this MRP by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

I understand that my MRP may be made electronically available to the public.

Smit Patel

# IMPROVING CUSTOMER CHURN PREDICTION THROUGH SEGMENTATION

Smit Patel

Master of Science 2023

Data Science and Analytics

Toronto Metropolitan University

## ABSTRACT

Customer churn prediction models have been proven to provide companies with the ability to determine if a customer will churn based on a variety of factors. However, this space still has room for improvement and this paper aims to determine if customer churn prediction accuracy can be improved using customer segmentation by cluster groups of customers based on similarities in their buying patterns and company engagement behaviors. Through this process, companies will also gain the ability to better understand which variables lead to a higher churn rate while also being able to use these clusters for targeted marketing initiatives.

**Key words:** Segmentation, Churn Prediction, DBSCAN, PCA, clustering

## Table of Contents

AUTHOR’S DECLARATION .....	ii
ABSTRACT .....	iii
<b>Section 1. Introduction</b> .....	1
<b>Section 2. Literature Review</b> .....	1-3
2.1. Customer Segmentation Techniques .....	2
2.2. Customer Churn Prediction Models .....	3
2.3. Evaluation and Validation Metrics .....	3
<b>Section 3. Methodology and Experiments</b> .....	4-16
3.1. Segmentation Techniques .....	4
3.2. Churn Prediction Techniques .....	5
3.3. Experiments .....	6
3.3.1. Dataset .....	6
3.3.2. Data Preprocessing .....	8
3.3.3. Outliers, Feature Importance & Balancing .....	9
3.3.4. Model Building – Segmentation .....	11
3.3.5. Model Building – Churn Prediction .....	16
3.3.6. Model Evaluation .....	16
<b>Section 4. Results</b> .....	17-19
4.1. Churn Prediction – No Segmentation .....	17
4.2. Churn Prediction with K Means .....	17
4.3. Churn Prediction with DBSCAN .....	18
4.4. Churn Prediction with GMM .....	19
<b>Section 5. Conclusions and Future Work</b> .....	19-20
<b>Section 6. References</b> .....	21-22

## List of Figures

Figure 1: Illustration of core points, border points and outliers .....	5
Figure 2: Count of Gender based on Churn Risk Score .....	8
Figure 3: Count of Feedback Type based on Churn Risk Score .....	8
Figure 4: Count of Membership Type based on Churn Risk Score .....	8
Figure 5: Correlation Plot for Numerical Variables .....	8
Figure 6: Boxplot for variables before outlier removal .....	9
Figure 7: Boxplot for variables after outlier removal .....	9
Figure 8: Feature Importance using RF Classifier .....	10
Figure 9: Plot of PCA Features and corresponding Variance % .....	11
Figure 10: K Means Elbow plot using KElbowVisualizer .....	12
Figure 11: Plot of clusters based on KMeans segmentation .....	12
Figure 12: Plot of clusters based on DBSCAN segmentation .....	14
Figure 13: Plot of clusters based on GMM segmentation .....	15
Figure 14: Correlation Plot for Gradient Boosting with DBSCAN .....	19

## List of Tables

Table 1: List of Variables with Variable Type .....	7
Table 2: Clusters from KMeans appended to dataset .....	13
Table 3: Clusters from DBSCAN appended to dataset .....	13
Table 4: Resulting Silhouette and Davies Bouldin Scores .....	15
Table 5: Clusters from DBSCAN appended to dataset .....	15
Table 6: Classification Report based on churn prediction .....	17
Table 7: Testing Accuracy for each given Model .....	17
Table 8: Testing Accuracy for models based on KMeans .....	18
Table 9: Testing Accuracy for models based on DBSCAN .....	18
Table 10: Testing Accuracy for models based on GMM .....	19
Table 11: Final Results of all clustering and prediction models .....	20

## Section 1: Introduction

In today's business world, companies are investing heavily in marketing to acquire new users. A survey found that companies are happy to allocate 57% of their budgets to digital marketing activities and plan to increase this spending by another 16% in 2023<sup>[1]</sup>. One prominent issue they face is that after such a significant initial investment, they would like to see an ROI (return on investment) and retain their customers over long periods of time to make the initial investment worthwhile. Further, it has also been found that acquiring a new customer is anywhere from 5- 25% more expensive than retaining an existing one<sup>[2]</sup> and increasing customer retention rates by just 5% increases profits from anywhere between 25% to 95%<sup>[3]</sup>. Anticipating and understanding customers' intentions to end a relationship is crucial for companies and can be considered a competitive advantage. This also applies to the e-commerce sector where it has been seen that there is a significant disconnect between companies and customers as there are so many options available to the consumer and they have the option of changing their purchasing pattern without informing sellers.

As the ability to predict customer churn already exists<sup>[11]</sup>, the goal of this paper is to identify if clustering customer data prior to predicting churn will increase prediction accuracy. The main reason behind being able to understand customer churn for a cluster of customers is to identify what patterns can be found for customers who churn from a company as well as those customers who do not churn from a company. Using this customer data and creating clusters of similar customers can allow a company to make decisions that will directly impact that cluster. For example, a company can identify customers who are likely to churn due to believing a product is overpriced and can then send this cluster a promotional offer (i.e., targeted marketing). In this research, customer data is first clustered using segmentation techniques including K-Means, DBSCAN, and GMM, and then used to build customer churn prediction models to determine churn prediction for each group of customers. The prediction accuracy of the clustered data is compared against the prediction accuracy over the overall dataset to conclude whether clustering data can improve customer churn prediction accuracy. After feeding customer data into the churn prediction classifier models of Random Forest, MLP, and Gradient Boosting, it was seen that the highest accuracy prediction model was the Gradient Boosting Classifier with a testing accuracy of 83%. Comparing this to the segmented customer data that was done using K-Means, DBSCAN, and GMM, it was seen that the testing accuracy did indeed improve and the combination of segmenting customer data with DBSCAN along with performing a churn prediction with the gradient boosting classifier yielded a testing accuracy of 93%.

## Section 2: Literature Review

The overall idea of predicting customer churn has been applied to industries ranging from telecommunications (to understand how customers act when in a contractual setting) to B2C and e-commerce (to understand how customers act when in a non-contractual setting where turnover is hard to determine). Regardless of the industry, the idea of companies keeping up with one another in a consumer-driven industry where options for a certain product are widespread and accessible to customers has also been explored to determine how companies can stand out to a customer<sup>[4]</sup>. The conclusion was that being able to predict customer churn for either a product or overall company as well as identifying crucial variables that affected a customer's reasoning to churn was of great importance to decrease overall customer churn<sup>[6]</sup>.

## 2.1 Customer Segmentation Techniques

There are a handful of studies that have explored how segmenting customers through different techniques can either improve or impair customer churn predictions. A research paper that looked to determine how customer segmentation affects customer churn predictions found two main factors that need to be considered. First, it is crucial to carry out segmentation when considering customer churn, and second, the prediction model being built requires variables that are highly correlated with churn scores as opposed to just performing a customer churn prediction on overall customer data<sup>[6]</sup>. So far, this has been done in the context of segmenting customers on time slices in intervals based on the given data, followed by predicting customer churn on the slices, which led to the conclusion that setting up multiple slices within a given dataset would improve performance in churn prediction<sup>[5]</sup>. This is a very informative discovery for businesses that collect data and improve their business based on timings, such as for e-commerce websites that look at how long consumers stay on given pages, how long it takes a consumer to check out, and how long the time interval is for a customer to return to the website to buy a product again. However, this is mainly important to B2C and e-commerce businesses, and would not have any impact on banks, telecommunication, and B2B companies that are not as focused on time interval data points.

Other clustering techniques have been introduced, including k-means, and the LRFM (length, recency, frequency, monetary) model<sup>[9]</sup>, both of which have made great contributions in segmenting customers for the purpose of churn prediction, with k-means being the most popular segmenting technique. However, there has been a missing component in the space as it has not been determined which segmenting technique leads to the best customer churn predictions. In addition to the two mentioned techniques, there are a handful of other clustering techniques that will be explored to determine which works best and leads to the highest accuracy churn prediction. Although k-means is the most used clustering technique, it does have its drawbacks in that the k-random initial centroids must be chosen prior to running the model. There are 2 methods that have been used to help determine what the ideal initial value should be. The first is to enumerate K from 2 to 8 and plot the results to obtain the relation between the contour coefficient and K to determine the number of clusters<sup>[6]</sup>. The second is to apply the k-means algorithm with different k values and plot the curves of the SSE and average silhouette coefficient against the number of clusters to analyze the two curves to identify the optimal number of clusters<sup>[7]</sup>. The optimal number of clusters can then be found by plotting the evaluation measure against the number of clusters and looking for the points in the plot at which a knee, peak, or dip forms.

Two other models used for the purpose of clustering large data are DBSCAN and GMM. Both are clustering methods revolving around clustering data based on distanced-based calculations, with each having its own individual benefits, which are discussed in Section 3.1. The main benefit to both DBSCAN<sup>[17]</sup> and GMM<sup>[16]</sup> is that they can process large and extremely large datasets to form meaningful clusters while reducing the computational effort required for each algorithm. This, in turn, can allow for more complex datasets to be parsed and iterations to be made for fine-tuning without computation power being a concern. Additionally, it was found<sup>[8]</sup> that both DBSCAN and GMM yield tighter boundaries while being able to recognize potential outliers better than K-Means, which is also of great value when working with large datasets.

As the use case for each of the clustering models expands beyond just being able to create meaningful clusters and involves being able to use these clusters to improve churn prediction, comparing the models against



each other will help to conclude which method truly yields the highest prediction accuracy. This is also compared to a scenario where no clustering is done to see how the results vary when a standard churn prediction is performed on an entire dataset.

## 2.2 Customer Churn Prediction Methods

Customer churn is an important metric that has been thoroughly explored in the data science space. From simple customer churn models involving regression models<sup>[11]</sup> to more complex machine learning models involving MLPs<sup>[17]</sup>, many methods have been explored to determine what the best fit for predicting customer churn is. Since a variety of methods have been researched already and this is not the target improvement of this paper, pre-existing customer churn prediction models with the highest accuracy will be used to determine if the test accuracy of these models improves when the added layer of customer segmentation is introduced.

In using pre-existing customer churn prediction models, the idea is to use a range of models that have high test accuracies. With this in mind, standard machine learning algorithms would be used, as well as ones that are more complex, and make use of subsets of machine learning such as neural networks to determine if the combination of each method along with segmentation improves the overall test accuracy. Starting with different standard ML algorithms and their effect on predicting customer churn, research<sup>[11]</sup> investigated LR, KNN, DT, SVM, RF, XGBoost, and LightGBM all being used against a customer dataset to predict customer churn with the highest accuracy. A handful of evaluation metrics were used to determine the validity of each model, and the results indicated that LR had the lowest accuracy at 82% while RF, LightGBM, and XGB with the highest accuracy at 89%, 88.95%, and 88.88% respectively<sup>[11]</sup>. With this information, it is decided that both Random Forest and XGB would be used as two of the customer churn prediction methods in this paper as they yield the highest test accuracy. Next, looking at models that make use of neural networks, research<sup>[10]</sup> found that using an MLP model with customer data yields a test accuracy of 94%, a momentum of 0.2, a learning rate of 0.3, and a total epochs of 500 using 6 hidden layers. As this test accuracy is even better than that of XGB and Random Forest, MLP would also be used in addition to the two, to determine if accuracy for all 3 models or a specific model increases in combination with the mentioned customer segmentation techniques.

## 2.3 Evaluation and Validation Metrics

The final and one of the most important steps through this research process is evaluation and validation, which involve using mathematical metrics to quantify the validity of a given model. The overall use case of evaluation metrics from a mathematical standpoint changes depending on the type of classification being done and the models being used. For binary classification, the most used identifiers involve looking at accuracy, misclassification, precision, recall, and f-score based on the false positive, true positive, true negative, and false negative predictions laid out by a model<sup>[14]</sup>. For rare event classification problems, such as in the case of customer churn, AUC (area under the curve) is the preferred accuracy performance measure to use<sup>[15]</sup>. As a result, AUC is one of the most common evaluation approaches used in this domain and is essentially equal to the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one and is one of the evaluation approaches used in this paper. As explained in another paper, an intuitive interpretation of the AUC is that it provides an estimate of the probability that a randomly chosen instance of class 1 is correctly rated

or ranked higher than a randomly selected instance of class 0 (i.e., the probability that a churner is assigned a higher probability to churn than a non-churner)<sup>[16]</sup>.

### Section 3: Methodology and Experiments

As explored in Section 2, the premise of this paper is to first cluster groups of data based on similarities, using common customer segmentation techniques including K-Means, DBSCAN, and GMM and towards a customer churn prediction. This will be compared to a simple customer churn prediction without any prior customer segmentation to determine if segmenting customers prior to performing a churn prediction increases test accuracy and leads to a better overall churn prediction. The segmentation techniques of K-Means, DBSCAN, and GMM were all chosen for the benefits mentioned in Section 2.1 in that they are the most common segmentation techniques used, they can efficiently cluster data from large to extremely large datasets while using less computational power and are not influenced or highly skewed by outliers in the dataset.

#### 3.1 Segmentation Techniques

The first and possibly most popular segmentation technique is K-Means, which essentially uses the distance between data points as a measure of ‘similarity’ to create clusters. This is done by starting with an arbitrary initialization value  $K$ , for the number of clusters that would be formed with the data and this  $K$  will be determined using methods discussed in Section 2.1 and serves as the initial centroids. From here, each data point  $X_n$  needs to be assigned to its closest cluster centroid  $C_n$ , which is done by calculating the distance between  $X_n$  and  $C_n$  and choosing a cluster associated with each data point where the distance between the data point and centroid is minimized. The distance metric used here is Euclidean distance and can be calculated as  $d(x, y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$ . This is done for each data point for the first iteration, and then the centroids are re-initialized by calculating the average of all data points for each cluster. This is done to determine the new cluster center and mathematically can be represented as  $C_i = \frac{1}{|N_i|} \sum x_i$ . This process is then repeated multiple times until the points stop moving and each centroid has been centered around a cluster, thus forming the final segmentations. Mini-Batch K-Means works in the same way in terms of determining clusters, however, it takes small randomly chosen batches of the dataset for each iteration, which proves to be useful when working with large datasets to process data easier. By working with data in batches, each update uses gradient descent and computational power can be heavily decreased.

The second segmentation technique is DBSCAN, which stands for Density-Based Spatial Clustering of Applications with Noise and its main benefit is that it can find arbitrarily shaped clusters and cluster with outliers, which is better for real-world applications where data is not as separable and easily clustered. DBSCAN operates with two parameters **eps**, which is the distance that specifies the neighborhoods where two points are neighbors if the distance between them is less than or equal to the **eps** parameter. The second parameter is **minPts**, which specifies the minimum number of data points to define a cluster. Using both parameters, points can be classified as core points, border points, or outliers, where core points are labeled if there are at least several points (specified by **minPts**) in an area with a radius specified by **eps**. Border points are classified if the point is reachable from a core point and there are less than a few points (specified by **minPts**). In Figure 1 below, red points are core points, yellow points are border points and blue points are outliers. In this way, the algorithm starts from a starting point and goes around marking the points as such, and once a cluster formation starts, all points within that

neighborhood of the initial point become a part of that cluster. From there, another point is randomly chosen from points that have not been visited and the same procedure is applied to form another cluster, which is repeated until all points have been visited once, thus creating the segmentations.

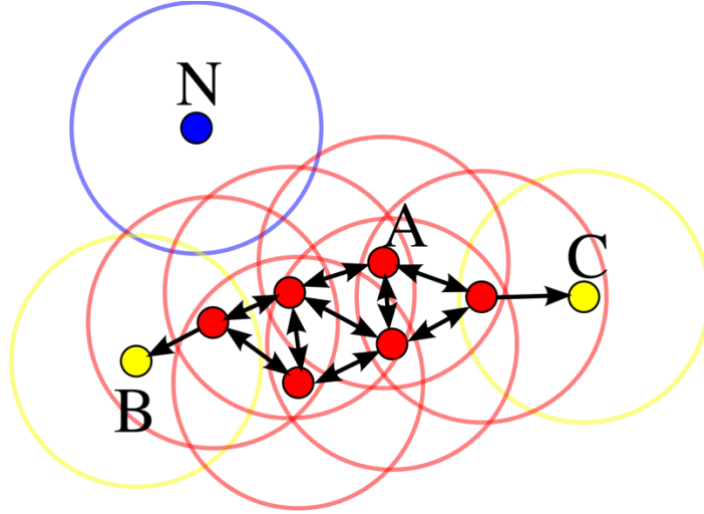


Figure 1 – Illustration of core points, border points and outliers<sup>[19]</sup>

The final segmentation technique in GMM, which stands for Gaussian Mixture Model and essentially follows the Expectation-Maximization (EM) algorithm, but for data points that follow a Gaussian distribution. The main benefit of GMM is that it can cluster data in any shape/form and go off the parameters of mean and covariance to describe the shape and position of each cluster. Since the model follows a Gaussian distribution, it can be represented mathematically as  $N(x; \mu, \Sigma) = \frac{1}{(2\pi\Sigma)^{1/2}} \exp(-\frac{1}{2}(x - \mu)^T \Sigma^{-1}(x - \mu))$  where  $x$  is the vector for a single data point,  $\mu$  is the vector for the mean for each cluster, and  $\Sigma$  represents the covariance matrix between each dimension. In terms of the algorithm itself, it works similarly to K-Means, where the number of clusters  $K$  is arbitrarily chosen, and the initial parameters for each centroid are randomly established. The difference comes in the part of forming clusters where the likelihood that the dataset originates within the initial set of clusters with the current mean and covariance parameters is calculated and this likelihood is then maximized by changing the parameters. This step is repeated until the change in the parameters is less than a set amount or until a fixed number of iterations is reached, thus forming the clusters in the necessary oblong clusters.

### 3.2 Churn Prediction Techniques

Upon completing a customer segmentation using the above methods, the next step is to use these segments to create churn prediction. This is done using Random Forest, MLP, and XGBoost to determine if there are any differences in accuracy between the two prediction techniques and to identify which prediction technique performs better both with and without a segmented dataset. The main reason for choosing each of these 3 churn prediction methods, as outlined in Section 2.2, is their ability to accurately predict customer churn with test accuracies higher than other machine learning classifier models.

Random Forest is a branch of decision trees and essentially builds a union of decision trees that are merged and trained with a bagging method to obtain an accurate prediction. In the sense of decision trees, each data point is evaluated against a certain condition amongst a set of criteria, and depending on the outcome of each check, the data point is routed through the tree to the next checkpoint. The criteria are identified during training by the algorithm, which creates the tree based on the training data set variable values. This decision-making and branching occurs until the algorithm reaches either the maximum tree depth (i.e., has gone through all the variables in the tree) or when no further split is possible. As mentioned before, where random forest differs is that it adds additional randomness to the model by training the decision tree on multiple datasets which are all overlapping subsets of the original dataset (i.e., the same data is sampled multiple times). This process is known as bagging where a random sample of data in a training set is selected with replacement.

MLP or multilayer perceptron's are machine learning models that make use of neural networks to generate a set of outputs based on a given set of inputs. As MLPs are multilayered, they consist of input nodes, output nodes, and multiple hidden layers that allow for processing and scaling data to train a network and arrive at one output when given multiple inputs. As an MLP is fully connected, each node of each layer is connected to every node in the succeeding layers by an associated weight value. In order to determine these weight values, MLPs make use of backpropagation, which is essentially a training process that requires taking the error rate between each node from the input node to the output node and feeding the loss to re-calculate the error rate in a backward manner from the output node to the input node while fine-tuning the weights of each layer in order to minimize the error.

XGBoost or Extreme Gradient Boosting can accurately make stronger predictions by combining the estimates of a set of simpler, weaker models. It uses a combination of both Gradient Boosting and Decision Trees where weights are assigned to all independent variables which are then fed into a decision tree that outputs a prediction. The weight of the variables that are classified as 'wrong prediction' by the tree is increased and these variables are fed to the subsequent decision tree. In this manner, each predictor can be compared against one another to produce a weighted average which results in an accurate and higher accuracy model. In terms of gradient boosting, the gradient descent algorithm is used to minimize the loss when adding new models. XGBoost would minimize a regularized objective function, L1 and L2, which combines a loss function based on the target- predicted output and a penalty term for model complexity. The training would then iteratively add new trees that predict the error term for prior trees, which are then all combined to come to a final prediction.

### **3.3 Experiments**

Now that the baseline of what each model is and how they will be implemented has been established, experiments can begin to be carried out to better understand how customer segmentation can be used to make more accurate churn predictions.

#### **3.3.1 Dataset**

The dataset used for this paper comes courtesy of an online data challenge<sup>[18]</sup>, consisting of data from a website as well as 20 variables of collected data for each individual customer who has held a membership on the website before. The dataset also offers a churn risk score variable ranging from 1 to 5, where 5 is most likely to churn. To better understand the variables in the dataset and what data cleaning needed to be done, some exploratory data analysis was done. The variables in the customer dataset can be seen in Table 1 with the first 7 being numerical variables, and the rest being categorical.

Variable	Variable Type
days_since_last_login	numerical - int
avg_time_spent	numerical - int
avg_transaction_value	numerical - int
avg_frequency_login_days	numerical - int
points_in_wallet	numerical - int
churn_risk_score	numerical - int
age	numerical - int
gender	categorical - object
region_category	categorical - object
membership_category	categorical - object
joined_through_referral	categorical - object
preferred_offer_types	categorical - object
medium_of_operation	categorical - object
Internet_option	categorical - object
used_special_discount	categorical - object
offer_application_preference	categorical - object
past_complaint	categorical - object
complaint_status	categorical - object
feedback	categorical - object

*Table 1 – List of Variables with Variable Type*

When looking at the categorical variables, there were a few observations that could be made. The first, which can be seen in Figure 2 for the number of males and females based on churn risk score, is that the dataset was overall balanced, which meant that no imputations would need to be done other than on the target variable (churn\_risk\_score) to balance the dataset. Additionally, based on the ‘Feedback’ variable that can be seen in Figure 3 below, most feedback that was given was negative and we can also see that those points always lead to a higher churn risk score (4-5), while the feedback that was positive lead to a lower churn risk score (between 1-3).

Lastly, based on the ‘Membership Type’ variable that can be seen in Figure 4 below, those customers with a higher tier membership type end up with a lower churn risk score, while those with a more standard or no membership end up with a higher churn risk score. Based on this, from a high level, it can be noted that these variables should be taken into consideration and can be considered more important for the skew that is created when the target variable is involved. Looking at numerical variables, a correlation plot was used to see if there were any numerical variables highly correlated with churn risk score, which can be seen in Figure 5 below. No variable is highly correlated with churn risk score, with ‘points\_in\_wallet’ and ‘avg\_transaction\_value’ displaying the highest correlation with a negative correlation of -0.29. Based on this, it can be noted that those two variables may be numerical variables of high importance as they are the highest correlated variables with the target variable.

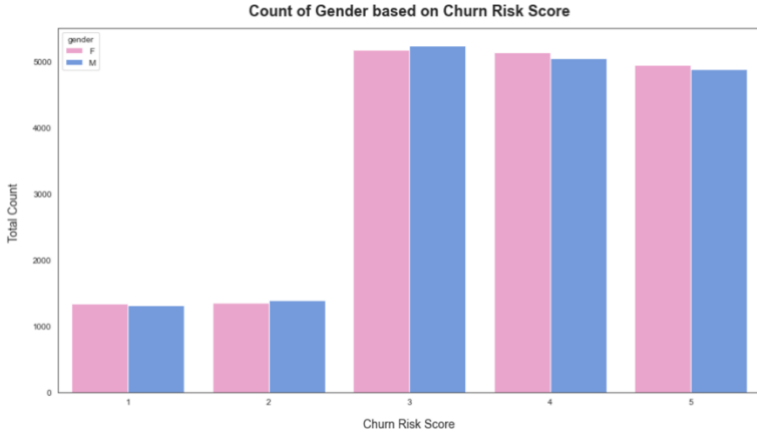


Figure 2 – Count of Gender based on Churn Risk Score

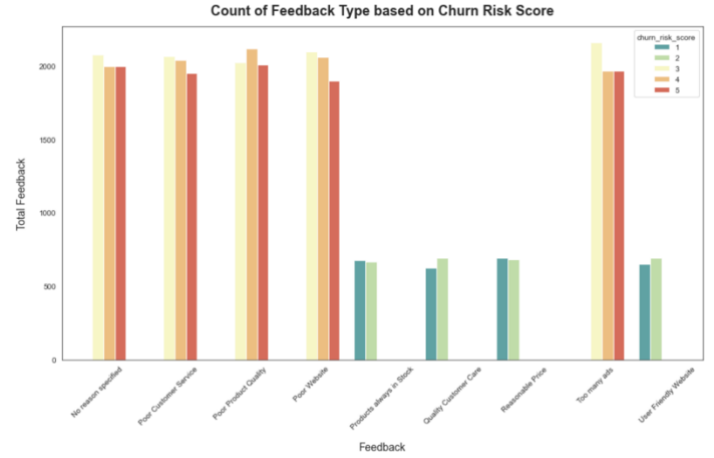


Figure 3 – Count of Feedback Type based on Churn Risk Score



Figure 4 – Count of Membership Type based on Churn Score

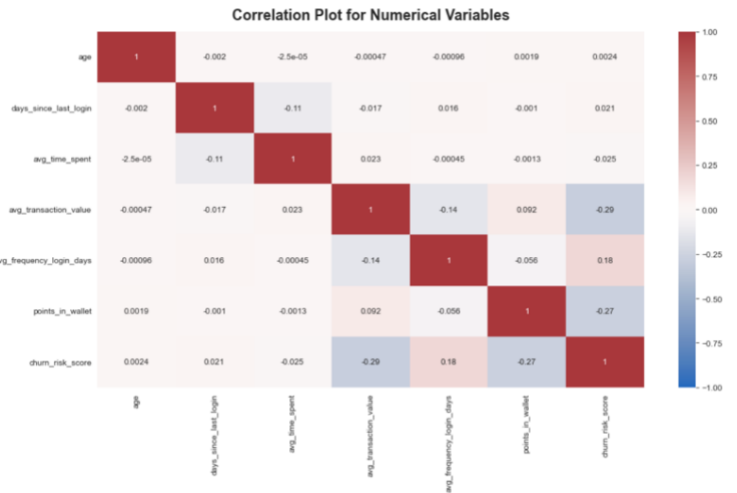


Figure 5 – Correlation Plot for Numerical Variables

### 3.3.2 Data Preprocessing

Prior to doing any actual modeling with the data, the preliminary step of pre-processing the data must be done to have a clean dataset that can produce valuable insights when plugged into a model. Starting with imputing values, there were many columns that either contained 'Unknown', '?', 'Error', or erroneous negative values that did not align with the variable. For the categorical variables 'gender', 'joined\_through\_referral', 'referral\_id', 'medium\_of\_operation', 'days\_since\_last\_login', each of the improper values was replaced with NaN and then the mode was taken and imputed for each of these variables. Based on research, it was found that mode was best for imputing categorical variables with a small number of unique values, which applied to this dataset. Next, for the numerical variables 'days\_since\_last\_login', 'avg\_time\_spent', 'points\_in\_wallet', and 'avg\_frequency\_login\_days', all erroneous and negative values were replaced with NaN and KNN imputation was done using sklearn's KNNImputer() method, which was found to be the preferred imputation method for numerical variables. To do so, a distance, k, is specified from the missing values, which are then predicted based on the mean of the neighbors. From here, some minor clean-up was done including adding a year variable based

on the joining\_date variable as well as taking out any negative churn risk (churn\_risk\_score of -1) from the dataset since this can be considered invalid data and cannot be imputed.

### 3.3.3 Outliers, Feature Importance & Balancing

Ideally when feeding data into a model, outliers would be minimized, the important features would be highlighted, and the target variable would be balanced for more accurate predictions to be made as the algorithms being used require classification data to have an equal number of observations for each class. Based on Figure 6 below, all numerical variables were observed using a boxplot to understand the distribution for each variable and determine which variables had outlier values. Looking at Figure 6, the numerical variables avg\_transaction\_value, points\_in\_wallet, avg\_time\_spent, and avg\_frequency\_login\_days all have outliers, and most outliers occur past the maximum of the boxplot. To handle these outliers, all the data points that are either less than  $Q1 - 1.5 * IQR$  or greater than  $Q3 + 1.5 * IQR$ , are removed from the dataset. IQR or interquartile range is simply the difference between quantile 3 and quantile 1, and the resulting boxplot after removing the outliers can be seen in Figure 7, where the total number of outliers has significantly decreased.

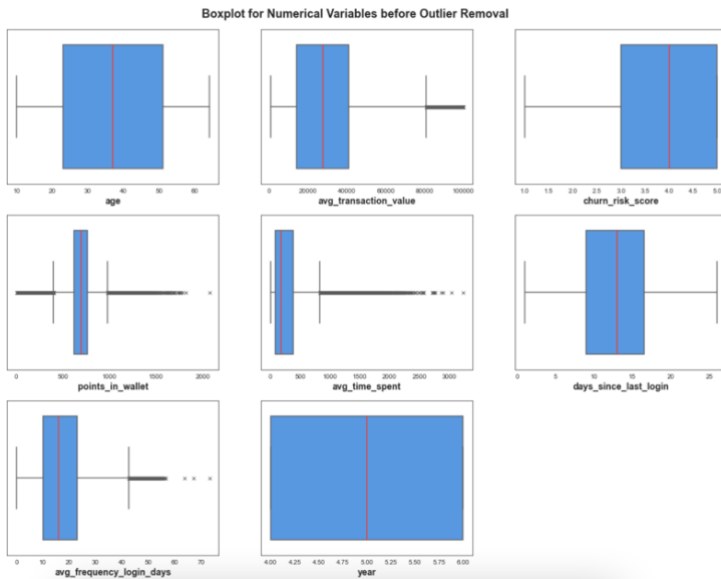


Figure 6 – Boxplot for variables before outlier removal

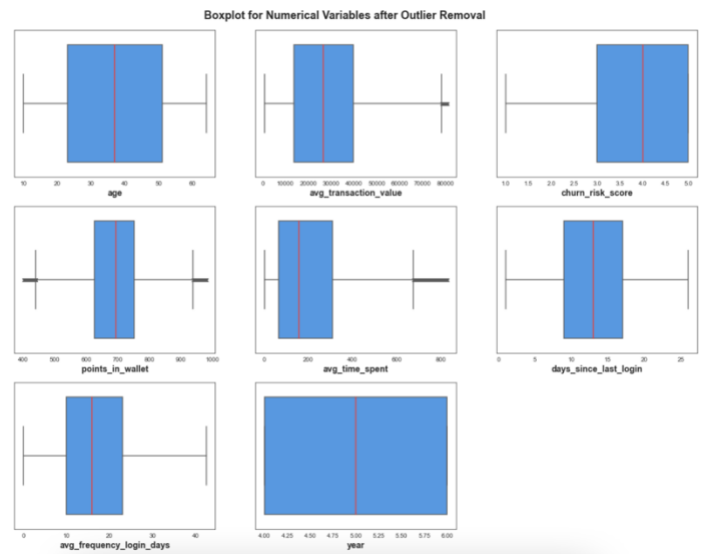


Figure 7 – Boxplot for variables after outlier removal

Looking at the value counts of the target label (churn\_risk\_score), each of the churn risk scores (1, 2, 3, 4, 5) were severely imbalanced where the churn risk scores of 1 and 2 only have a count of  $\sim 2700$  data points, while the churn risk scores of 3, 4, 5 all have over 10,000 data points. To handle this, oversampling is used where a minority class is specified, and oversampling is done to either duplicate data points in the minority class or synthesize new data points from existing data points in the minority class. Specifically, the SMOTE or Synthetic Minority Oversampling Technique is used, which balances the class distribution by randomly increasing the minority class data points by replicating them). This is done using SMOTE() method as part of the imblearn package, which takes the values of a dataset as input where the dataset contains the numerical features as well as dummies for each of the categorical features. Upon doing this, the RobustScaler() method is used from the sklearn package to re-scale the dataset and remove any additional outliers that have been generated through the SMOTE

process. Next, the categorical data goes through one hot encoding, and dummy variables are created to convert the categorical variables into numeric, which would allow for them to be used with the clustering models.

Finally, identifying the important features of the dataset is crucial to understanding which features/variables will have a larger effect on both the clustering and churn prediction models and in turn selecting a subset of variables for the training that will lead to both improved efficiency and effectiveness of the predictive model. To form this subset for the churn prediction model, we use the pre-existing conclusions generated in Section 3.3.1 as well as the RandomForestClassifier() method from the sklearn package. This function essentially fits an X\_train and Y\_train variable that are developed by taking a train/test split based on setting the features and target variables as well as the random state. This model can then be visualized using a simple bar plot through Matplotlib and the resulting plot can be seen in Figure 8 below. Based on this plot, it can be seen that the variables that are scored/ranked as the highest importance in relation to churn\_risk\_score are points\_in\_wallet, avg\_transaction\_value, membership\_category\_Basic Membership, membership\_category\_No Membership, avg\_frequency\_login\_days, avg\_time\_spent, days\_since\_last\_login, and each of the membership categories. We can conclude that these variables will contribute most to a customer exiting/churning from the company's services, so a new and final dataset is formed for the rest of this research with these variables used. Looking at the customer segmentation, the idea is to use variables related to demographics as well as variables that are not used towards the churn prediction model to increase the variance for each of the customer segmentation models to create more meaningful clusters. Upon performing exploratory data analysis, it was decided that the variables gender\_F, gender\_M, region\_category\_City, region\_category\_Town, region\_category\_Village, past\_complaint\_No, past\_complaint\_Yes, joined\_through\_referral\_No, and joined\_through\_referral\_yes would be used for the customer segmentation.

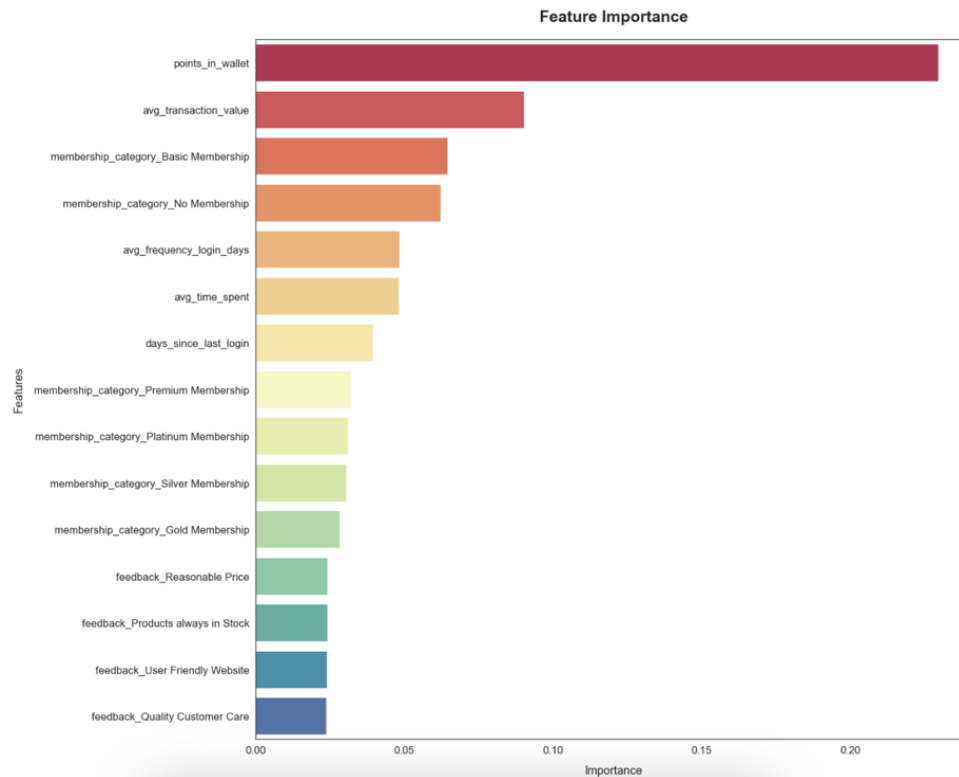


Figure 8 – Feature Importance using RF Classifier



### 3.3.4 Model Building - Segmentation

For the segmentation piece, K-Means, DBSCAN, and GMM are used as discussed in Section 3.1. There are two-fold pre-processing steps necessary for each method, which is the same for each method. The first fold is to standardize all the columns of the dataset, which essentially removes the pre-existing mean and scales each feature/variable to unit variance. The purpose of this is so that each variable has an equal contribution to its effect on model fitting and to avoid any biases being created through the prediction process and can be done using the `StandardScaler()` function provided by `sklearn.preprocessing`. The second fold is to feed and fit the scaled features into a PCA model, to model the data in a smaller subset of variables that can be analyzed and visualized throughout the process, and to come up with insights of actual importance. Within this subset, the data is also retained while keeping the data as linearly separable as possible to form clusters. This process is done using the PCA model from `sklearn.decomposition` and involves setting an arbitrary number of components (in this case 5) and fitting the newly created scaled features dataset along the range of components. The resulting plot of variance % for each PCA feature can be seen in Figure 9, and the highest variance of features is in columns 0, 1, and 2, so these will be fed into each segmentation model, to allow the models to minimize the variance and create clusters that would be more effective when used in the churn prediction models. The variance % greater than 0.175 is used towards the customer segmentation model.



Figure 9 –Plot of PCA Features and corresponding Variance %

Starting with K-Means, the first step is to understand what value of  $k$  to use for the k-means model, and this can be done by observing the elbow score for each  $K$ , where the idea is to choose a  $K$  value located at the ‘elbow joint’ index of the plot. Using the `KElbowVisualizer` function from the `yellowbrick` package, the  $k$  range is set as between 1 to 50, and the PCA features are fit to the model. The resulting plot can be seen in Figure 10, where the ‘elbow joint’ can be seen at  $K = 7$ , which would be the number of clusters specified in the `KMeans` model for this data. A second metric used to evaluate the effectiveness of the `KMeans` algorithm is also employed which is the silhouette score from the `sklearn.metrics` package. After the PCA data is fit to the `KMeans` model with 7 clusters and an arbitrary set `random_state`, the silhouette score is evaluated using the Euclidean distances. The silhouette score essentially outlines how similar samples within a cluster are to each other, thus describing how efficient the clusters are. Values for silhouette score range between -1 to 1 and the closer the score is to 1, the more dense and well-separated a cluster is distinguished to be. The clusters are then appended to the initial dataset to further group the dataset by each cluster.

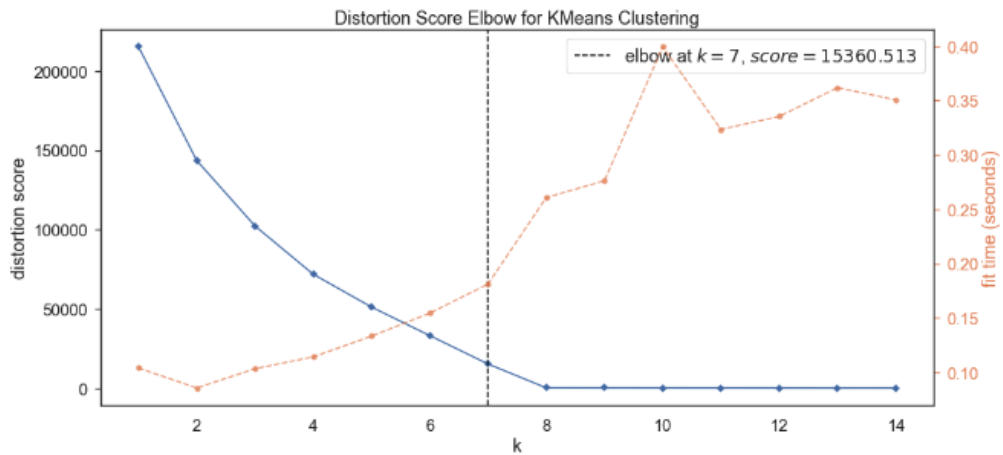


Figure 10 – K Means Elbow plot using KElbowVisualizer.

Looking at Figure 11, it can be seen what values are assigned to each cluster. For example, Cluster 5 contains data points with the highest churn\_risk\_score and negative feedback and the lowest points\_in\_wallet and no positive feedback making Cluster 5 of high value in identifying factors for high churn customers. In contrast, Cluster 13 contains data points with the lowest churn\_risk\_score and negative customer feedback with the highest points\_in\_wallet and positive customer feedback, making it of most value in identifying factors for low churn customers.

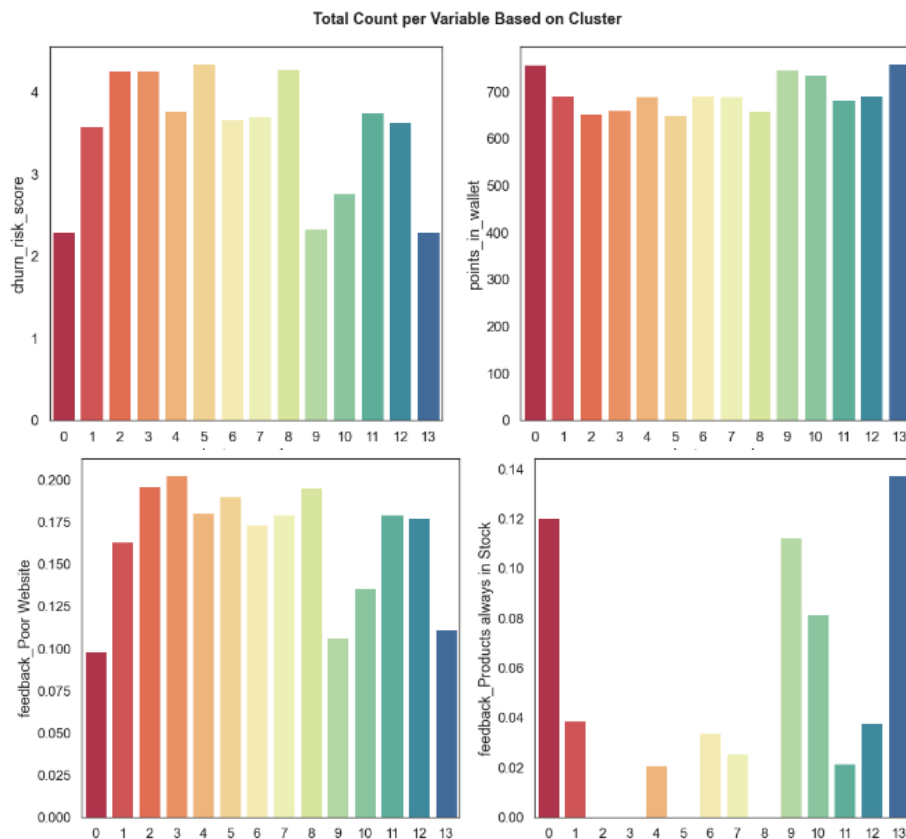


Figure 11 – Plot of clusters based on KMeans segmentation.

After taking the created clusters and adding them to the initial training dataset (seen in Table 2), the dataset is then grouped into separate subsets of datasets based on the cluster number (i.e., cluster 0 would be one dataset, cluster 1 would be another, etc.) and the grouped dataset would then go through the SMOTE balancing method and churn prediction process outlined in sections 3.3.3 and 3.3.5 to understand if clustering customer data using KMeans prior to performing a churn prediction improves the accuracy or not.

feedback_No reason specified	feedback_Poor Customer Service	feedback_Poor Product Quality	feedback_Poor Website	feedback_Products always in Stock	feedback_Quality Customer Care	feedback_Reasonable Price	feedback_Too many ads	feedback_User Friendly Website	cluster
0	0	0	0	1	0	0	0	0	5
0	0	0	0	0	1	0	0	0	2
0	0	0	1	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0

Table 2 – Clusters from KMeans appended to dataset.

Next, DBSCAN is employed using the DBSCAN() model from sklearn.cluster package. This model requires only 2 parameters eps and min\_samples where eps define the neighborhood of a point, x, and min\_samples are the minimum number of points in a neighborhood needed in order to form a dense region. To determine what values to use for these parameters, the NearestNeighbors function is used from the sklearn.neighbors package, which essentially involves fitting the PCA data to the model and calculating the distance and index between each data point. Then sorting and plotting the results in descending order allows for the same ‘elbow joint’ technique employed for KMeans to be used to determine the ‘elbow joint’ point, which is used for eps. Doing this yields an eps of 0.8 and a min\_samples of 9, which are input into the model and fit to create a new set of clusters. This set is then appended to the initial training dataset as dbscan\_cluster and looking at Table 3 below, the clusters assigned to each row based on KMeans differ from the assignment of clusters by DBSCAN.

feedback_Poor Customer Service	feedback_Poor Product Quality	feedback_Poor Website	feedback_Products always in Stock	feedback_Quality Customer Care	feedback_Reasonable Price	feedback_Too many ads	feedback_User Friendly Website	cluster	dbscan_cluster
0	0	0	1	0	0	0	0	5	0
0	0	0	0	1	0	0	0	2	1
0	0	1	0	0	0	0	0	0	2
0	0	1	0	0	0	0	0	0	2
0	0	1	0	0	0	0	0	0	2

Table 3 – Clusters from DBSCAN appended to dataset.

Furthermore, looking at Figure 12 below, we can once again see how the clusters are chosen where the separation is not as exaggerated with a normal distribution. In this case, clusters 4 and 6 are very similar in that they have the lowest churn\_risk\_score, avg\_time\_spent, and negative feedback while also having the highest positive feedback. Cluster 6 has higher positive feedback and would be of greatest value in terms of understanding customers' habits with low churn rates. The dataset is then grouped into separate subsets of datasets based on the cluster number and the grouped dataset would then go through the SMOTE balancing method and churn prediction process outlined in sections 3.3.3 and section 3.3.5 to understand if clustering customer data using KMeans prior to performing a churn prediction improves the accuracy or not.

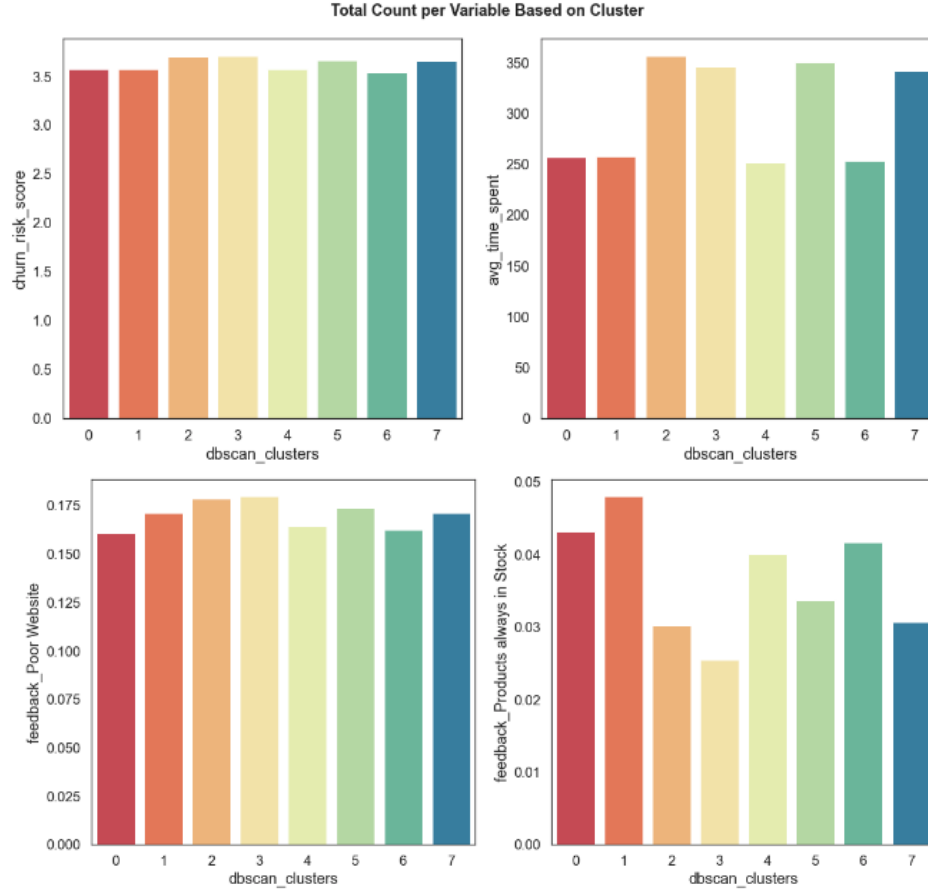


Figure 12 – Plot of clusters based on DBSCAN segmentation.

Lastly, GMM is employed using the GaussianMixture function from the sklearn.mixture package, which requires parameters `n_components`, `covariance_type`, `max_iter` and `random_state` to be identified. Both `max_iter` and `random_state` are arbitrary values and `max_iter` is set to a large number (in this case 2000), while `random_state` is arbitrarily set to 5. To determine both `n_components` and `covariance_type`, a for loop is used to iterate through the PCA data, for the covariance types of full, tied, diag, spherical, and number of clusters ranging from 1 to 15. For each iteration, the silhouette score and Davies Bouldin score are calculated with the objective of having the highest silhouette score as well as the lowest Davies Bouldin score. After running the for loop, the values are sorted in descending order based on silhouette score, resulting in the most optimal covariance type to be diag, and the number of clusters to be 8, which yields a silhouette score of 0.956 and Davies Bouldin score of 0.061. The results of this sorting can be seen in Table 4. These parameters are then fed through the GaussianMixture model with the PCA data fitted to it, and the new set of GMM clustered are produced and appended to the original training dataset. Looking at Table 5 below, it can once again be seen that the GMM clusters assigned to each row are different from those of KMeans and DBSCAN.

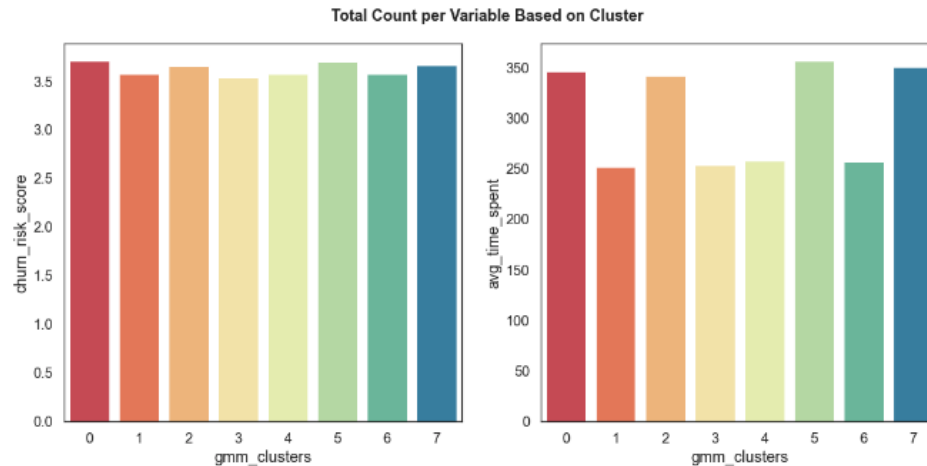
	Covariance Type	# of Clusters	Silhouette Score	Davies Bouldin Score
26	diag	8	0.958816	0.081612
6	full	8	0.958816	0.081612
36	spherical	8	0.958816	0.081612
16	tied	8	0.958816	0.081612
27	diag	9	0.884132	0.268688
7	full	9	0.883876	0.302182
17	tied	9	0.883876	0.302182

Table 4 – Resulting Silhouette and Davies Bouldin Scores

feedback_Poor Product Quality	feedback_Poor Website	feedback_Products always in Stock	feedback_Quality Customer Care	feedback_Reasonable Price	feedback_Too many ads	feedback_User Friendly Website	cluster	dbscan_cluster	gmm_cluster
0	0	1	0	0	0	0	5	0	5
0	0	0	1	0	0	0	2	1	3
0	1	0	0	0	0	0	0	2	0
0	1	0	0	0	0	0	0	2	0
0	1	0	0	0	0	0	0	2	0

Table 5 – Clusters from GMM appended to dataset.

Looking at Figure 13, it can also be seen that the created clusters are once again not as exaggerated and are slightly more normally distributed, similar to DBSCAN. For the final time, the dataset is then grouped into separate subsets based on the cluster number, and the grouped dataset then goes through the SMOTE balancing method and churn prediction process outlined in sections 3.3.3 and section 3.3.5 to understand if clustering customer data using KMeans prior to performing a churn prediction improves the accuracy or not.



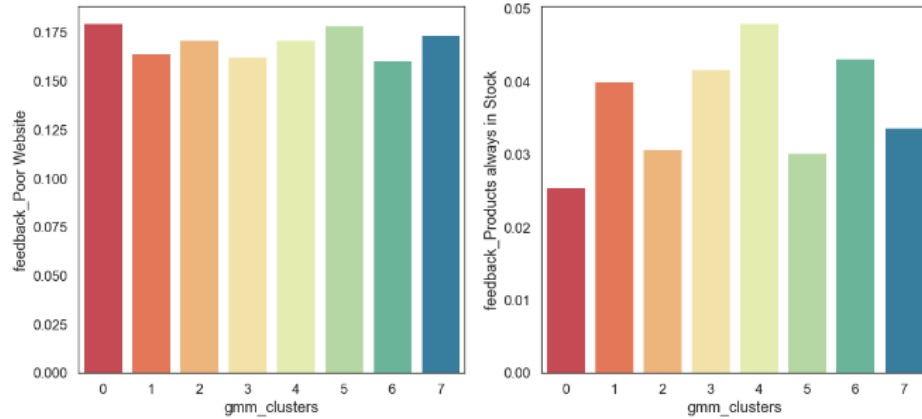


Figure 13 – Plot of clusters based on GMM segmentation.

### 3.3.5 Model Building – Churn Prediction

For the churn prediction, a Random Forest Classifier, MLP Classifier, and a Gradient Boosting Classifier are used. To start, the data was split as 70% towards the training set, and 30% towards the testing set. Starting with the Random Forest Classifier, an arbitrary high value is set for `n_estimators` (in this case 100), which specifies the number of trees in the forest of the model as well as `max_depth` which is the number of splits that each decision tree is allowed to make. Setting this value too low leads to the model underfitting the data and setting it too high leads to the model overfitting the data. The general values used for `max_depth` are 3, 5, or 7, but as this dataset is quite large, a higher value of 25 was chosen as the value for this parameter. Using both parameters, the `x_train` and `y_train` specified in the split are fed into the `RandomForestClassifier` model from `sklearn`, and a `y_prediction` is created for each `x_test` instance. Next, the MLP classifier is utilized where the number of hidden layers and nodes needs to be specified under `hidden_layer_sizes`. The general rule of thumb for datasets with large dimensions/features is to use 3-5 hidden layers and with this in mind, 3 hidden layers are chosen. To determine the number of nodes in each layer, the general equation used is  $\sqrt{(\text{input layers nodes} * \text{output layer nodes})}$ , and as each subsequent layer requires a decreased number of nodes in order to identify the target class, the nodes selected are 120, 95, and 70. The parameters of `x_train` and `y_train` are now fed into the `MLPClassifier` model from `sklearn` and a `y_prediction` is created for each `x_test` instance. Finally, a gradient boosting classifier is used with 4 different parameters specified. The first two are `n_estimators` and `max_depth`, which are kept the same as in random forest for uniform results. The other two parameters are `min_sample_leaf` and `min_samples_split`, which define the minimum observations required in a terminal leaf and the minimum observations required in a node to be considered for splitting. Using cross-validation, these are tuned to be set to 4 and 5. Once again, both the `x_train` and `y_train` parameters are fed into the `GradientBoostingClassifier` model from `sklearn` and a `y_prediction` is created for each `x_test` instance.

### 3.3.6 Model Evaluation

For each churn prediction method, a classification report is generated based on the `y_test` and `y_prediction` variables as well as an f1 score which is an evaluation metric that measures a model's accuracy by combining the precision and recall scores of a model. Both are generated using the `sklearn.metrics` package and are then compared against each other for each churn prediction model to compare the train accuracy, test accuracy, and test f1-score to determine which model performs best in terms of test accuracy. This metric would be the main evaluation metric in understanding which model performs best. In combination with both the

segmented and non-segmented datasets, it would also help conclude which combination of segmentation method and churn prediction method yields the highest test accuracy and thus would be most useful in improving customer churn predictions.

## Section 4. Results

### 4.1 Churn Prediction – No Segmentation

For the first churn prediction, no segmentation was done to build a baseline for what the outcome of a standard churn prediction would be with no segmentation of customer data. Looking at Table 6, an example of how the classification report prints the results of the random forest classifier prediction over 5 iterations can be seen where the final accuracy was 0.82 with an f1 score of 0.815. A final table with the test accuracies and test f1 scores based on each of the prediction models is then created and can be seen in Table 7 below. MLP classifier prediction had a final test accuracy of 0.81 with an f1 score of 0.808, while the gradient boosting classifier prediction had a final accuracy of 0.83 with an f1 score of 0.833. Comparing all 3 results below, Gradient Boosting is the ideal and most accurate model for predicting customer churn without any segmentation involved.

	precision	recall	f1-score	support
1	0.86	0.91	0.89	3072
2	0.91	0.86	0.88	3145
3	0.89	0.93	0.91	3175
4	0.75	0.51	0.61	3138
5	0.71	0.90	0.79	3106
accuracy			0.82	15636
macro avg	0.82	0.82	0.82	15636
weighted avg	0.82	0.82	0.82	15636

0.8153817403030639

Table 6 – Classification Report based on churn prediction.

	Model	Train Accuracy	Test Accuracy	Test f1-Score
0	Random Forest	1.000000	0.822077	0.815382
1	MLP	0.987282	0.809350	0.808776
2	Gradient Boosting	1.000000	0.835252	0.833922

Table 7 – Testing Accuracy for each given Model

### 4.2 Churn Prediction with K-Means

Upon completion of all setups of the KMeans model in Section 3.3.4, the silhouette score is calculated and a value of 0.857 is obtained, which confirms that the model was deployed in a way the clusters are of actual value. After running all 3 classifier prediction models on each individual cluster, the weighted average of all clusters is taken and a final table of each of the test accuracies and test f1-scores is created as can be seen in Table 8. Based on Table 8, the Random Forest classifier prediction had a final test accuracy of 0.87 with a f1 score of 0.86, the MLP classifier prediction had a final test accuracy of 0.804 with a f1 score of 0.804, while the gradient

boosting classifier prediction had a final accuracy of 0.847 with a f1 score of 0.844. Comparing all 3 results below, when using K-means to cluster customer data before doing a segmentation, the test accuracy of both Random Forest and Gradient Boosting increased, while the accuracy of MLP stayed the same. The test accuracy of Random Forest increased by 5%, while Gradient Boosting increased by 1%. It can be concluded that KMeans in combination with Random Forest will yield the best results for predicting customer churn and that clustering data prior does indeed increase test accuracy for these three models.

	Prediction Model	Training Accuracy (%)	Testing Accuracy (%)	Testing f1-score (%)
0	Random Forest	1.000000	0.870050	0.860888
1	MLP	1.000000	0.804455	0.804899
2	Gradient Boosting	1.000000	0.847772	0.844289

Table 8 – Testing Accuracy for models based on KMeans.

### 4.3 Churn Prediction with DBSCAN

Upon completion of all setups of the DBSCAN model in Section 3.3.4, all 3 classifier prediction models were run, and the results can be seen in Table 9. The Random Forest classifier prediction had a final test accuracy of 0.925 with a f1 score of 0.922, the MLP classifier prediction had a final test accuracy of 0.92 with an f1 score of 0.918, while the gradient boosting classifier prediction had a final accuracy of 0.934 with an f1 score of 0.932. Comparing all 3 results below, when using DBSCAN to cluster customer data before doing a segmentation, the test accuracy of all 3 models saw the biggest increase between 10-12% and all 3 models produced a higher test accuracy than K Means clustering. Out of the 3 models, Gradient Boosting yielded the highest test accuracy, and it can be concluded that DBSCAN in combination with Gradient Boosting will yield the highest test accuracy for predicting customer churn and would be the recommended method for future churn prediction. The confusion matrix can be seen below in Figure 14, where the predicted vs. actual labels are displayed for each churn risk score based on the PCA labels that were applied. Here it can be seen that the model performed best when predicting label 2 with 0 incorrect predictions!

	Prediction Model	Training Accuracy (%)	Testing Accuracy (%)	Testing f1-Score (%)
0	Random Forest	1.000000	0.925287	0.922891
1	MLP	1.000000	0.920690	0.918279
2	Gradient Boosting	1.000000	0.934483	0.932420

Table 9 – Testing Accuracy for models based on DBSCAN.



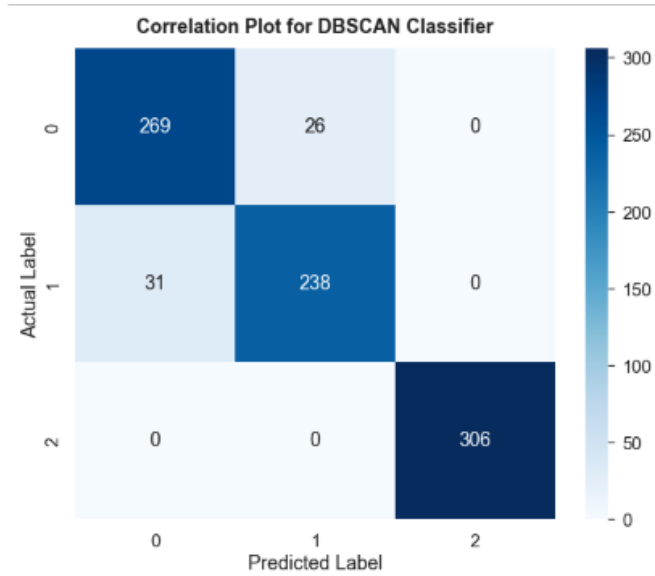


Figure 14 – Correlation Plot for Gradient Boosting with DBSCAN

#### 4.4 Churn Prediction with GMM

Upon completion of all setups of the GMM model in Section 3.3.4, all 3 classifier prediction models are run, and the results can be seen in Table 10 below. The Random Forest classifier prediction had a final test accuracy of 0.851 with a f1 score of 0.841, the MLP classifier prediction had a final test accuracy of 0.239 with a f1 score of 0.266, while the gradient boosting classifier prediction had a final accuracy of 0.845 with an f1 score of 0.843. Comparing all 3 results below, when using GMM to cluster customer data before doing a segmentation, the test accuracy of both Random Forest and Gradient Boosting increases by 2-3% while MLP has significantly dropped and can be deemed unusable in this scenario. Out of the 3 models, Random Forest yields the best results in conjunction with GMM.

	Prediction Model	Training Accuracy (%)	Testing Accuracy (%)	Testing f1-score (%)
0	Random Forest	1.000000	0.851613	0.841816
1	MLP	0.239748	0.266276	0.153665
2	Gradient Boosting	1.000000	0.845161	0.843894

Table 10 – Testing Accuracy for models based on GMM.

## Section 5. Conclusions and Future Work

After going through the process of testing each combination of customer segmentation techniques along with each churn prediction technique, the combination of DBSCAN with Gradient Boosting is the most accurate and effective method producing a test accuracy of 93%. It can also be concluded that performing a churn prediction on an entire dataset will only provide test accuracy scores between 80-85% while segmenting the data into clusters beforehand can lead to approximately a 10% increase in test accuracy. The results can be seen in Table 11 below.

Clustering Model	Prediction Model	Train Accuracy	Test Accuracy	Test f1-Score
None	RF	1.000000	0.822977	0.815382
None	MLP	0.987282	0.809350	0.808776
None	GBoosting	1.000000	0.835252	0.833922
KMeans	RF	1.000000	0.870050	0.860888
KMeans	MLP	1.000000	0.804455	0.804899
KMeans	GBoosting	1.000000	0.847772	0.844289
DBSCAN	RF	1.000000	0.925287	0.922891
DBSCAN	MLP	1.000000	0.920690	0.918279
DBSCAN	GBoosting	1.000000	0.934483	0.932420
GMM	RF	1.000000	0.851613	0.841816
GMM	MLP	0.239748	0.266276	0.153665
GMM	GBoosting	1.000000	0.845161	0.843894

Table 11 – Final Results of all clustering and prediction models

These results are promising and can help companies with large customer-based datasets to better understand what groups of customers are more likely to churn. By clustering customers into groups of similarity, companies can get a better understanding of what factors are either deterring or positively impacting customers to churn or not churn from using their product and can make business decisions accordingly. This study was performed using website data based on a single product and may need to be altered to be applied to companies with multiple catalogs of products or with collected data that differs from the mentioned. In terms of future work, there is still room for improvement as the churn prediction models can still be tweaked and improved. Through more research and iterative work, improving all 3 classifier models (Random Forest, MLP, and Gradient Boosting) through parameter tuning could lead to even better accuracy than what was displayed in this report.

## Section 6. References

- [1] Moorman, C., Soli, J., & Cardoso, D. (2022, July 29). *Closing the Gap Between Digital Marketing Spending and Performance*. Harvard Business Review. <https://hbr.org/2022/07/closing-the-gap-between-digital-marketing-spending-and-performance>
- [2] Gallo, A. (2014, October 29). *The value of keeping the right customers*. Harvard Business Review. <https://hbr.org/2014/10/the-value-of-keeping-the-right-customers>
- [3] Reichheld, F. (2001). *Prescription for cutting costs*. Bain & Company. [https://media.bain.com/Images/BB\\_Prescription\\_cutting\\_costs.pdf](https://media.bain.com/Images/BB_Prescription_cutting_costs.pdf)
- [4] Aydin, S., & Özer, G. (2005). *The analysis of antecedents of customer loyalty in the Turkish mobile telecommunication market* [Review of *the analysis of antecedents of customer loyalty in the Turkish mobile telecommunication market*].
- [5] Gattermann-Itschert, T., & Thonemann, U. W. (2021). *How training on multiple time slices improves performance in churn prediction* [Review of *How training on multiple time slices improves performance in churn prediction*].
- [6] Xiahou, X., & Harada, Y. (2022). *B2C E-Commerce Customer Churn Prediction Based on K-Means and SVM* [Review of *B2C E-Commerce Customer Churn Prediction Based on K-Means and SVM*].
- [7] Rachid, A. D., Abdellah, A., Belaid, B., & Rachid, L. (2018). *Clustering Prediction Techniques in Defining and Predicting Customers Defection: The Case of E-Commerce Context* [Review of *Clustering Prediction Techniques in Defining and Predicting Customers Defection: The Case of E-Commerce Context*].
- [8] Xiao, T., Wan, Y., Jin, R., Qin, J., & Wu, T. (2022). *Integrating Gaussian Mixture Dual-Clustering and DBSCAN for Exploring Heterogeneous Characteristics of Urban Spatial Agglomeration Areas* [Review of *Integrating Gaussian Mixture Dual-Clustering and DBSCAN for Exploring Heterogeneous Characteristics of Urban Spatial Agglomeration Areas*].
- [9] Arthur Middleton Hughes. (2012). *Strategic database marketing: the masterplan for starting and managing a profitable, customer- based marketing program*. Mcgraw-Hill Professional.
- [10] Hudaib, A., Dannoun, R., Harfoushi, O., Obiedat, R., & Faris, H. (2015). *Hybrid Data Mining Models for Predicting Customer Churn* [Review of *Hybrid Data Mining Models for Predicting Customer Churn*]. In *International Journal of Communications, Network and System Sciences*.
- [11] Elyusufi, Y., & Kbir, M. (2022). *Churn Prediction Analysis by Combining Machine Learning Algorithms and Best Features Exploration* [Review of *Churn Prediction Analysis by Combining Machine Learning Algorithms and Best Features Exploration*]. In *International Journal of Advanced Computer Science and Applications*.
- [12] Keramati, A., Jafari-Marandi, R., Aliannejadi, M., Ahmadian, I., Mozaffari, M., & Abbasi, U. (2014). *Improved churn prediction in telecommunication industry using data mining techniques* [Review of *Improved churn prediction in telecommunication industry using data mining techniques*]. In *Applied Soft Computing* (Vol. 24, pp. 994–1012).
- [13] Weiss, G. M. (2004). *Mining with rarity: a unifying framework* [Review of *Mining with rarity: a unifying framework*]. In *ACM SIGKDD Explorations Newsletter* (Vol. 6, pp. 7–19).
- [14] Verbeke, W., Dejaeger, K., Martens, D., Hur, J., & Baesens, B. (2012). *New insights into churn prediction in the telecommunication sector: A profit driven data mining approach* [Review of *New insights into churn prediction in the telecommunication sector: A profit driven data mining approach*]. In *European Journal of Operational Research* (Vol. 218, pp. 211–229).
- [15] Li, K., Ma, Z., Robinson, D., & Ma, J. (2018). *Identification of typical building daily electricity usage profiles using Gaussian mixture model-based clustering and hierarchical clustering* [Review of *Identification of typical building daily electricity usage profiles using Gaussian mixture model-based clustering and hierarchical clustering*]. In *Applied Energy* (Vol. 231, pp. 331–342).
- [16] Chen, Y., Zhou, L., Bouguila, N., Wang, C., Chen, Y., & Du, J. (2021). *BLOCK-DBSCAN: Fast clustering for large scale data* [Review of *BLOCK-DBSCAN: Fast clustering for large scale data*]. In *Pattern Recognition* (Vol. 109).

- [17] Amuda, K. A., & Adeyemo, A. B. (2019). *Customers Churn Prediction in Financial Institution Using Artificial Neural Network* [Review of *Customers Churn Prediction in Financial Institution Using Artificial Neural Network*].
- [18] *Predict the churn risk rate | Practice Problems*. (n.d.). HackerEarth. Retrieved August 29, 2023, from <https://www.hackerearth.com/problem/machine-learning/predict-the-churn-risk-rate-11-fb7a760d/>.
- [19] *DBSCAN*. (n.d.). Wikipedia. Retrieved August 29, 2023, from <https://en.wikipedia.org/wiki/DBSCAN#/media/File:DBSCAN-Illustration.svg>