Predicting Customer Churn Using Logistic Regression

Group 11, Section MSA/A

Alexandra Pfleegor, Hardik Patel, Anushka Sarda
Tae Eun (Harrison) Kwon, Muhamad Imannulhakim

December 1, 2023

1 Introduction

Subscription-based business models often have one goal leading their decisions: how can they keep customers using their service? The formal word for this problem is churn, defined by Merriam Webster to be "a regular, quantifiable process or rate of change that occurs in a business over a period of time as existing customers are lost and new customers are added" [1]. Thus, companies want to pursue strategies to reduce their customer churn.

In the world of streaming services, these strategies could include having the best entertainment available on their site, as well as competing with other streaming services in terms of price.

Our project centers on a California-based telecommunications company. This company provides both internet and TV services to its consumers. We want to know: what characteristics are most correlated with a customer ending their subscription? Where should the company direct their marketing efforts to retain customers?

2 Problem Statement

In this project, we aim to predict the probability of customer churning based on independent variables such as demographics, service usage, billing, customer engagement and more. We will identify variables that have a significant influence on customer churn and if there are specific customer demographics or subscription plans that have varying churn rates.

From this analysis, we will be able to identify factors that have a significant impact on customer churn. This analysis would allow this Californian Telecommunications company to identify subscription plans that lead to customer churn or identify customer demographics whom are likely to churn, and prioritize their marketing strategy to retaining those customers.

The variables we will be analyzing fall into the following broad categories: customer demographics, subscription plans, service usage, and billing. Due to the large quantity of variables provided in the dataset, our main focus during preprocessing will be to reduce the size of the dataset, and ensuring the sample data that we will use for analysis is clean, complete, and accurate. After identifying the best preprocessing method, we will create a model that best fits the dataset and allows us to predict customer churn.

Our overarching goal for this project is to construct a robust logistic regression model that estimates the probability of customer churn. Through this analysis, we intend to identify the key factors influencing churn rates and pinpoint specific customer demographics and subscription plans associated with higher churn rates. This valuable insight will empower the Californian Telecommunications company to strategically allocate resources and prioritize marketing efforts towards retaining at-risk customers, ultimately enhancing customer retention and business success in a highly competitive industry.

3 Data Description

We got our data from a Kaggle dataset including information on all 7,043 customers from a Telecommunications company in California for the second quarter of 2022 [2]. Thus, all data points are from a single moment in time. Not including customer ID, there are 37 variables in the dataset including 23 qualitative and 14 quantitative variables. These variables are described in tables 1 (qualitative) and 2 (quantitative).

Table 1: Qualitative Variables

Variable	Categories
Gender	Female, Male
Married	Yes, No
City	1106 categories, will not use
Zip Code	1626 categories
Offer	None, Offer A, Offer B, Offer C, Offer D, Offer E
Phone Service	Yes, No
Multiple Lines	Yes, No, None[1]
Internet Service	Yes, No
Internet Type	Cable, Fiber Optic, DSL, None[2]
Online Security	Yes, No, None[2]
Online Backup	Yes, No, None[2]
Device Protection Plan	Yes, No, None[2]
Premium Tech Support	Yes, No, None[2]
Streaming TV	Yes, No, None[2]
Streaming Movies	Yes, No, None[2]
Streaming Music	Yes, No, None[2]
Unlimited Data	Yes, No, None[2]
Contract	Month-to-Month, One Year, Two Year
Paperless Billing	Yes, No
Payment Method	Credit Card, Bank Withdrawl, Mailed Check
Customer Status	Stayed, Churned, Joined
Churn Category	Competitor, Dissatisfaction, Attitude, Price, Other, None
Churn Reason	See note for categories[*]∗

[1] where Phone Service = No

[2] where Internet Service = No

[*] Competitor had better devices, Competitor made better offer, Attitude of support person, Don't know, Competitor offered more data, Competitor offered higher download speeds, Attitude of service provider, Price too high, Product dissatisfaction, Network reliability, Long distance charges, Service dissatisfaction, Moved, Extra data charges, Limited range of services, Poor expertise of online support, Lack of affordable download/upload speed, Lack of self-service on Website, Poor expertise of phone support, Deceased, or no reason given.

Since there are too many variables for our analysis, we first discard those that would either be difficult to use or irrelevant to our analysis, including Customer ID, City, Zip Code, Latitude, Longitude, Churn Category and Churn Reason. For Churn Category and Churn Reason, although the data could be helpful for our analysis, this data is only included for those customers who stopped using the company during the second quarter of 2022. Further research could look at these variables and create a classification model with the data to predict whether a customer will leave and if so, their reasoning.

The histogram plots showing the count of data points in each category, separated by customer status, do not show any strong patterns for Married, Age, or Gender as seen in figure 1. It is difficult to tell, but it seems like the Number of Dependents variable shows some pattern: people with no dependents have a higher chance of churning than people with dependents. We will not know this for sure until we look at the regression analysis.

The next set of histogram plots, shown in figure 2, do not show a strong pattern for Offer or Phone Service. However, there seems to be a correlation between Number of Referrals and Customer Status as well as Tenure in Months and Customer Status. In both cases, the likelihood of churning seems to decrease as the values increase. This would make sense, since people who are just starting out with the company could be trying out different phone or TV plans. Also, if someone referrals many people, we can assume that they

Table 2: Quantitative Variables

Variables
Age
Number of Dependents
Latitude
Longitude
Number of Referrals
Tenure in Months
Avg Monthly Long Distance Charges
Avg Monthly GB Download
Monthly Charge
Total Charges
Total Refunds
Total Extra Data Charges
Total Long Distance Charges
Total Revenue

like their current telecommunications company.

The sixth set of histogram plots, found in figure 6 in the appendix, show a correlation between Contract and Customer Status. This also makes sense because people who are locked into a one or two year contract cannot cancel their service as easily as someone whose contract is month-to-month. The other variables in the histograms (Unlimited Data, Paperless Billing and Payment Method), however, do not appear to have a strong correlation with Customer Status.

The rest of the sets of histogram plots do not seem to show a strong correlation between Customer Status and each independent variable. These sets can be found in figures 3 (Avg Monthly Long Distance Charges, Multiple Lines, Internet Service, Internet Type), 4 (Avg Monthly GB Download, Online Security, Online Backup, Device Protection Plan), 5 (Premium Tech Support, Streaming TV, Streaming Movies, Streaming Music), 7 (Monthly Charges, Total Charges, Total Refunds, Total Extra Data Charges), and 8 (Total Long Distance Charges, Total Revenue).

Finally, we graphed box plots for all numerical variables against Customer Status, as seen in figures 9, 10, and 11. One notable observation from these plots is that the means of Tenure in Months seem to be significantly different from one another between customers who left and those who stayed. Other potentially notable plots are the ones with Total Long Distance Charges and Total Revenue. It is difficult to tell whether the means of the customers who stayed are significantly different from those who left. We can also see that there seem to be a large number of outliers in the data that could potentially skew the results. Below, we will talk about how we dealt with these outliers.

4 Analyses

4.1 Data Pre-Processing

After loading the data set, we conducted an initial exploration to identify any problems with the data. We noticed that many of the columns had missing values, particularly in the 'Avg Monthly Long Distance Charges,' 'Multiple Lines,' 'Internet Type,' 'Avg Monthly GB Download,' and various service-related columns. The variable names and the percent of null values is displayed in figure 12. To ensure our model is accurate and reliable, we used a systematic approach based on the context of the data to fill in the missing values. For numerical columns, we filled missing values with zero, assuming that customers with missing data did not incur additional charges or use certain services. For categorical columns related to internet services, we filled missing values with 'None,' indicating that the customer did not have internet service.

Additionally, we identified columns related to specific services, such as 'Online Security,' 'Online Backup,' 'Device Protection Plan,' 'Premium Tech Support,' 'Streaming TV,' 'Streaming Movies,' 'Streaming Music,' and 'Unlimited Data.' For customers without internet service, we filled missing values in these columns with 'No,' indicating the absence of these services. The process of filling missing values was carried out systematically to maintain the integrity of the data and prepare it for subsequent analysis. Details on the updated null percentages after imputation are located in figure 13.

To keep our focus on predicting customer churn, we filtered out rows where 'Customer Status' was 'join' or 'joined.' We were left with 6,589 rows after the filtering process. Along with that, we dropped certain columns that deemed irrelevant or challenging to use for our predictive model as well as the columns that had over 70% null values. The excluded columns include 'Customer ID,' 'City,' 'Zip Code,' 'Latitude,' 'Longitude,' 'Churn Category,' and 'Churn Reason.

As outliers can negatively impact he performance of predictive models, we applied the Adjusted Interquartile Range (IQR) method to identify and remove outliers. We excluded certain columns, such as 'Number of Dependents,' 'Number of Referrals,' 'Total Refunds,' and 'Total Extra Data Charges,' from this outlier removal process, as these columns may naturally exhibit variations. We identified 155 outliers and had 6,434 rows left. At this point, we randomly sampled 1500 rows to use for the data analysis process.

Categorical variables need to be converted into numerical representations, as regression models operate on numerical data. We encoded categorical variables using the OneHotEncoder technique. This process transforms categorical columns into binary vectors, making them suitable for regression. For example, Gender_Male is 1 when Male and 0 when Female.

We then applied scaling to the numerical variables using 'StandardScaler.' This is a common preprocessing step to ensure features are on a similar scale, preventing certain features from dominating others during model training.

4.2 Checking for Multicollinearity

To understand the relationships between independent variables, we created a correlation matrix and visualized it using a heatmap shown on 14. The heatmap visually represents the correlation between each pair of variables. High correlation coefficients suggest potential multicollinearity issues. It can be seen that there are many darker areas that suggest multicollinearity. We also created variance inflation factors (VIF) to determine highly correlated variables. The VIF values can be found in figure 15. Using a threshold of 10, We found that the following columns in table suggested multicollinearity and removed them from the analysis (VIF values greater than 10 could suggest multicollinearity).

Table 3: Omitted Variables due to High Correlation

Monthly Charge	Tenure in Months	Total Refunds	Total Extra Data Charges
Total Long Distance Charges	Total Revenue	Internet Service_Yes	Internet Type_None

This updated matrix shown in figure 16 reflects a reduction in correlation coefficients between features showing the successful mitigation of multicollinearity. The updated VIF values after the removal of the variables can be found on figure 17. The complete table with precise values can be found in figure 7.

4.3 Variable Selection

After processing our data, we needed to decrease the number of features in our model using variable selection. There are two reasons to limit the number of variables: to avoid overfitting and to reduce the complexity of our model. The two main techniques we chose were Lasso Regression and Stepwise Regression.

Lasso regression aims to reduce the number of variables by limiting the sum of the absolute value of the coefficients to some number T. The model uses that "budget" of T first on the most important variables (most predictive), forcing other variables to (or close to) 0.

In our first iteration of Lasso regression, we split our model into a training set and a testing set. After finding the best value of α ("the penalty term that denotes the amount of shrinkage (or constraint) that will be implemented" [3]), we fit our training data to the Lasso regression model. The results from our optimized model are shown in table 8. We then sorted the variables based on the absolute value of their coefficients before choosing the best 15 features. Thus, the Lasso model chose the variables seen in table 4.

Next, we tried both variations of stepwise regression: forward selection and backward elimination. In forward selection, the model starts with no factors and iteratively adds in factors based on whether the model

Table 4: Lasso Regression Variables

Contract_Two Year	Contract_One Year	Offer_Offer E	Number of Referrals	Payment Method_ Credit Card
Age	Number of Dependents	Total Charges	Paperless Billing_Yes	Internet Type_Fiber Optic
Streaming TV_Yes	Offer_Offer D	Online Security_Yes	Married_Yes	Internet Type_DSL

improves in a specific measure like AIC. In our model, using the python function SequentialFeatureSelector, the model tries to maximize the cross-validation score in each step. Backward elimination, conversely, starts with a model with all factors. It then iteratively finds the worst factor based on whether the model improves and removes it.

We again split our data into a training and testing set before running forward selection. In this case, we did not give the model a specific number of variables. Thus, the forward selection model chose the variables shown in table 5.

Table 5: Forward Selection and Backward Elimination Variables

Streaming Movies_Yes	Offer_Offer A	Offer_Offer B	Offer_Offer E
Internet Type_Fiber Optic	Married_Yes	Contract_One Year	Contract Two_Year
Paperless Billing_Yes	Age	Number of Referrals	

However, after performing backward elimination, we got the same variables as in table 5.

Studying both tables, we can see that both methods chose the variables Contract_One Year, Contract_Two Year, Number of Referrals, Age, Paperless Billing_Yes, Married_Yes, Internet Type_Fiber Optic, and Offer_Offer E. Thus, there is a good amount of overlap between the two tables, which means that both methods agreed that the above variables are important predictors. However, we will still fit both models to determine which set of predictors are more significant.

4.4 Model Fitting

In this project, we used logistic regression, since it estimates the probability of something happening (like customer churn).

First, we used the variables from our Lasso regression model in a logistic regression model with Customer Status as the dependent variable. The results can be seen in the Appendix in table 9. In this result, all variables except Online Security_Yes and Internet Type_DSL are significant. Since Internet Type_Fiber Optic has a small p-value, the category Internet Type is statistically significant. Further calculations find that the recall value is 0.71, AUC-ROC is 0.91 and accuracy is 0.84. **make table with this and refer to it**.

Next, we fit our logistic regression model with the variables chosen by stepwise regression. The results can be seen in table 10 in the Appendix. All variables in this regression model are significant except Married_Yes, Offer_Offer A, and Offer_Offer B. However, since Offer E is significant, the category of Offer must be a statistically significant predictor. With this model, we get a recall value of 0.69, AUC-ROC of 0.89 and accuracy of 0.83.

Before moving on to accuracy and goodness of fit discussions, we should also find the results for the full model (using all of the variables) for comparison. The results of this regression can be found in table 11 in the Appendix. More variables are insignificant in this model. The significant (at the $\alpha = 0.05$ level) predictors are Married_Yes, Offer_Offer A, Offer_Offer D, Offer_Offer E, Internet Type_Fiber Optic, Streaming TV_Yes, Contract_One Year, Contract_Two Year, Paperless Billing_Yes, Payment Method_Credit Card, Age, Number of Dependents, Number of Referrals, and Total Charges.

For both of our reduced models, we then performed tests seeing whether the reduced model was even better than the full model. For both, here are our hypotheses:

$$H_0 : \text{the coefficients of the additional predictors in the full model are all equal to 0}$$
$$H_A : \text{at least one coefficient of the additional variables in the full are is not equal to 0}$$

In other words, our null hypothesis states that the reduced model is better than the full model, while the alternative states that the additional predictors in the full model significantly improves the model. After finding the maximization of the likelihood function under the reduced and full models, we get a deviance test statistic of 22.64 for the lasso based model and 64.62 for the stepwise regression based model.

For the lasso based model, this corresponds to a p-value of 0.2050. Thus, we fail to reject the null hypothesis, meaning that the reduced model is, at the 95% confidence level, better than the full model.

On the other hand, for the stepwise regression based model, the corresponding p-value is 0.0000, meaning that we reject the null hypothesis. This indicates that there is strong evidence to reject the null hypothesis and conclude that the full model (with all predictors) is a significantly better fit for the data than the reduced model (choosing just the variables from stepwise regression).

Table 6: Comparing Models

Model	**Accuracy**	**Recall**	**AUC-ROC**
Logistic Regression Model with Lasso variable selection	0.843333	0.714286	0.910142
Logistic Regression Model with stepwise variable selection	0.826667	0.692308	0.888164

When comparing our two models with metrics such as accuracy, recall and AUC-ROC (seen in table 6), we can compare their performances. Even though we already know that the stepwise based logistic regression model is not better than the full model, we will still need to evaluate these metrics for comparison.

Both models have similar accuracy and AUC-ROC values. However, the logistic regression model with Lasso regularization has a slightly higher accuracy (0.8433 vs. 0.8267) and a slightly higher recall (0.7143 vs. 0.6923) compared to the logistic regression model with stepwise feature selection (stepwise). Based on these metrics alone, the Lasso model seems to perform slightly better.

However, before concluding that the lasso based logistic model is our final model, we need to perform some goodness of fit tests to check our assumptions.

4.5 Goodness of Fit Checks

Our hypotheses for the goodness of fit tests are as follows:

$$H_0 : \text{the logistic model fits the data}$$
$$H_A : \text{the logistic model does not fit the data}$$

For the logistic regression model based on Lasso variable selection, the p-value from the deviance test is 1.0, meaning that we fail to reject the null hypothesis. So, our model is a good fit to the data. Similarly, with a p-value of 0.99377, the logistic regression model based on stepwise regression variable selection is also a good fit to the data.

4.6 Optimization and Tuning

After choosing our model, we wanted to tune the hyper-parameters to improve our metrics. Specifically, we wanted to look at the threshold for predicting 1 or 0 with logistic regression. Originally, we just used 0.5 in order to compare models, but we could improve by choosing a different threshold.

For all threshold values from 0.01 to 1.00, we found the accuracy and recall scores and plotted them on our graph shown in figure 18. We then looked at two different points on the graph: the threshold where recall is approximately equal to accuracy (0.3) and the threshold that maximizes accuracy (0.43). At the threshold of 0.3, accuracy decreases to 0.8133, while recall increases to 0.8242 from the original model with a threshold of 0.5. At the 0.43 threshold, accuracy increases from the original model to 0.8533, while recall increases to 0.8022. although the recall is slightly better at the 0.3 threshold, the accuracy decreases more than the increase in recall.

We thus decided to use a threshold of 0.43 for our final model. Both recall and accuracy increase from our original model, improving our model quality metrics.

4.7 Validation

When testing our models on test data before, we first split the test data into validation and testing data. We used the test data for finding the best model between our two options and then used the validation data to find the unbiased evaluation of our model. After testing on our validation data, the results are a recall of 0.70, AUC-ROC of 0.89 and accuracy of 0.81. Therefore, we did not overfit our model to our original data.

5 Conclusions and Recommendations

In our final model, shown in table 9, the model selected 12 predictors that are considered statistically significant. These predictors are associated with customer churn, and their coefficients provide insights into the direction and strength of the relationships. Notable predictors include:

- Contract type (Two Year and One Year): Customers with longer contract durations (1-2 years) are less likely to churn than customers with month-to-month contracts, as indicated by negative coefficients.
- Offer type (Offer E): Customers targeted with Offer E are more likely to churn than those who are not given an offer, as indicated by a positive coefficient.
- Number of referrals: More referrals are associated with a lower likelihood of churn (negative coefficient).
- Payment Method (Credit Card): Customers using credit card payment methods are less likely to churn (negative coefficient).
- Internet Type (Fiber Optic): Customers with Fiber Optic internet are more likely to churn (positive coefficient).

Longer contracts may provide customers with more stability and commitment, making them less likely to switch to a different service provider. It's common for businesses to offer incentives, discounts, or special deals to encourage customers to commit to longer contracts as a way to reduce churn. Having your payment method as credit card means that it is most likely automatically charged each month. Thus, unless the customer carefully looks, increases in price are not as noticeable.

Our analysis has identified key predictors of customer churn and provided valuable insights for the Californian Telecommunications company. To enhance customer retention, we recommend further customer segmentation to tailor marketing strategies, optimizing offers through A/B testing, and incentivizing referrals to reduce churn. Additionally, promoting longer contract durations and credit card payment methods can help improve customer stability. Special attention should be given to Fiber Optic internet customers, and exploring the reasons behind their higher churn rates is essential.

Future actions should include real-time predictive analytics, continuous customer feedback collection, competitor analysis, and a deeper dive into churn categories and reasons. By implementing these recommendations, the company can proactively address churn and work towards improving customer retention and overall satisfaction.

6 Appendix

List of Figures

List of Tables

Listings

A Data Description

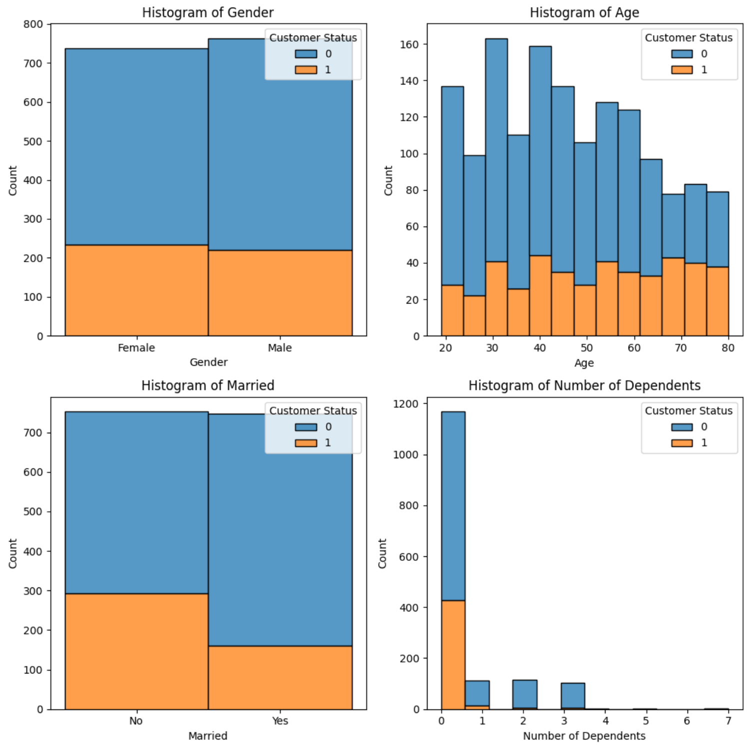

Figure 1: Histogram Plots for Gender, Age, Married and Number of Dependents

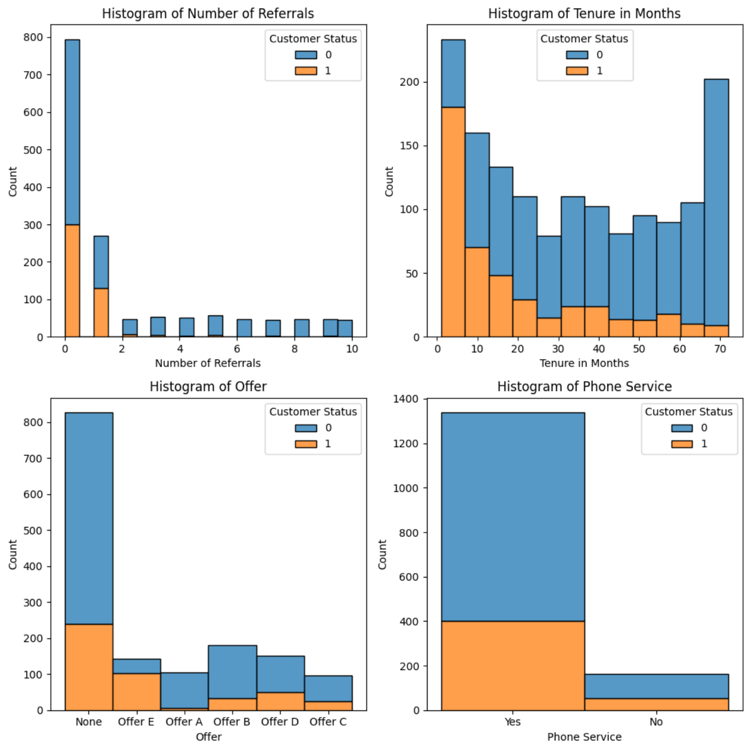

Figure 2: Histogram Plots for Number of Referrals, Tenure in Months, Offer and Phone Service

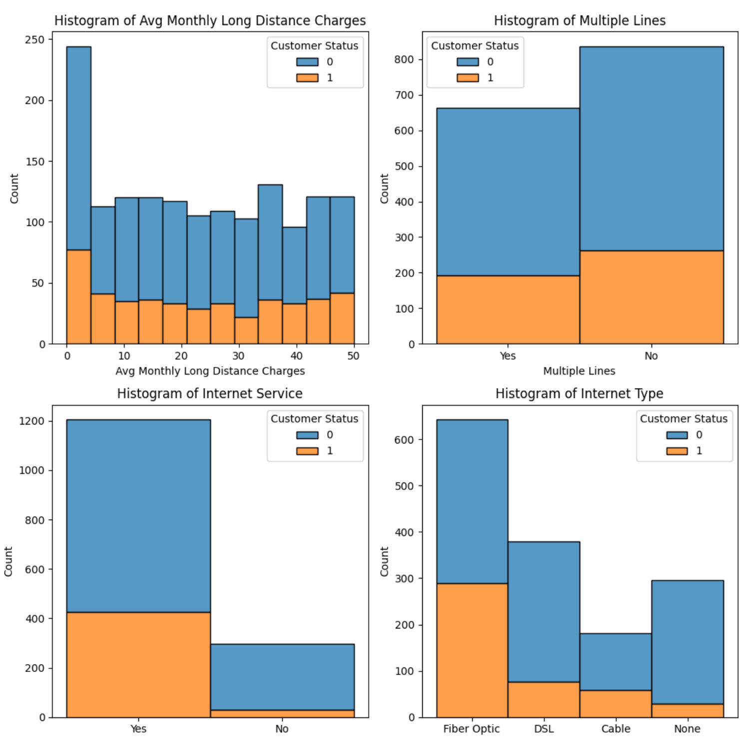

Figure 3: Histogram Plots for Avg Monthly Long Distance Charges, Multiple Lines, Internet Service and Internet Type

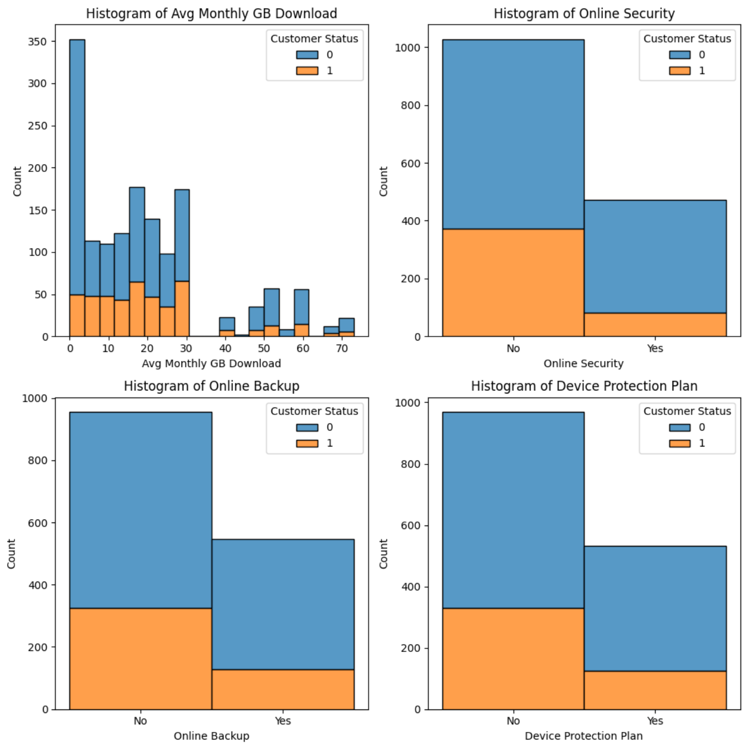

Figure 4: Histogram Plots for Avg Monthly GB Download, Online Security, Online Backup and Device Protection Plan

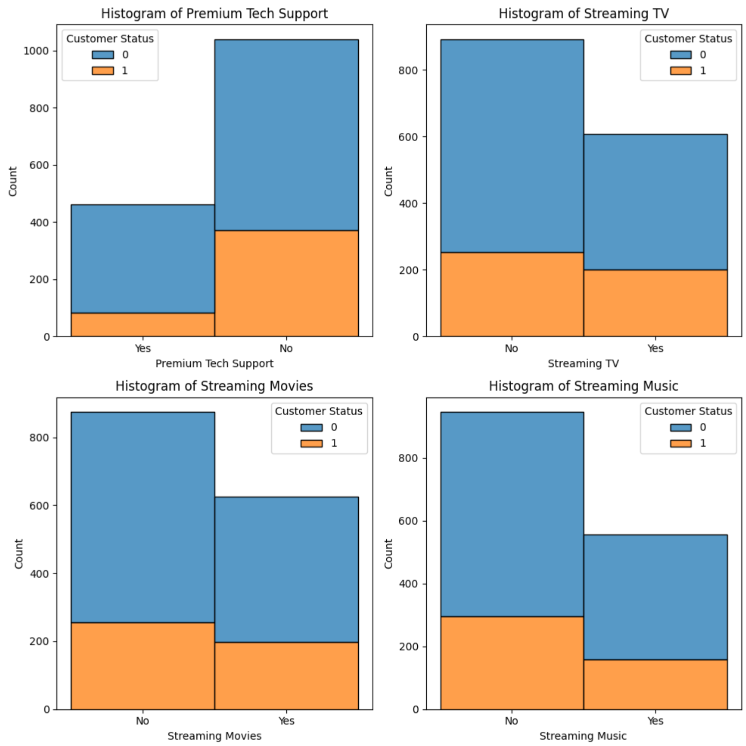

Figure 5: Histogram Plots for Premium Tech Support, Streaming Tv, Streaming Movies and Streaming Music

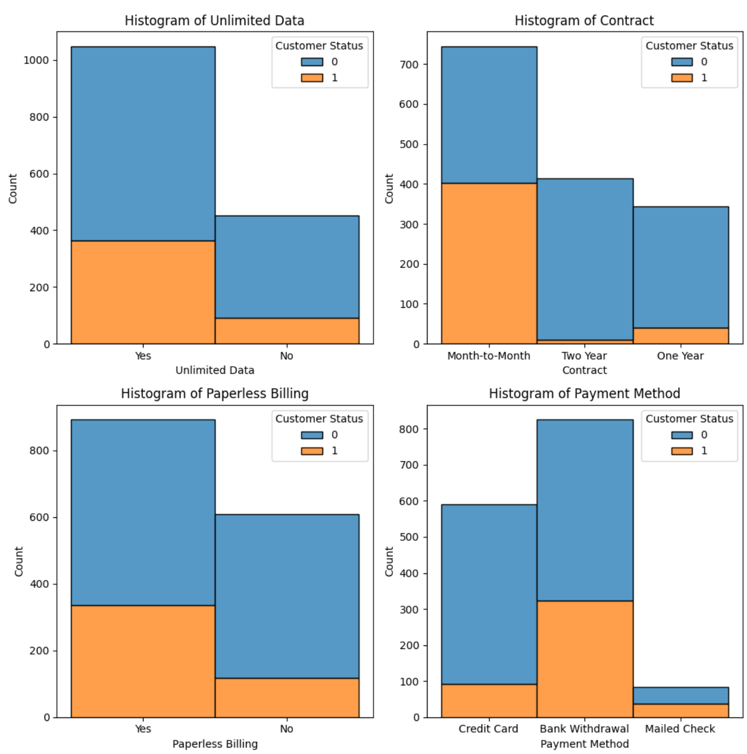

Figure 6: Histogram Plots for Unlimited Data, Contract, Paperless Billing and Payment Method

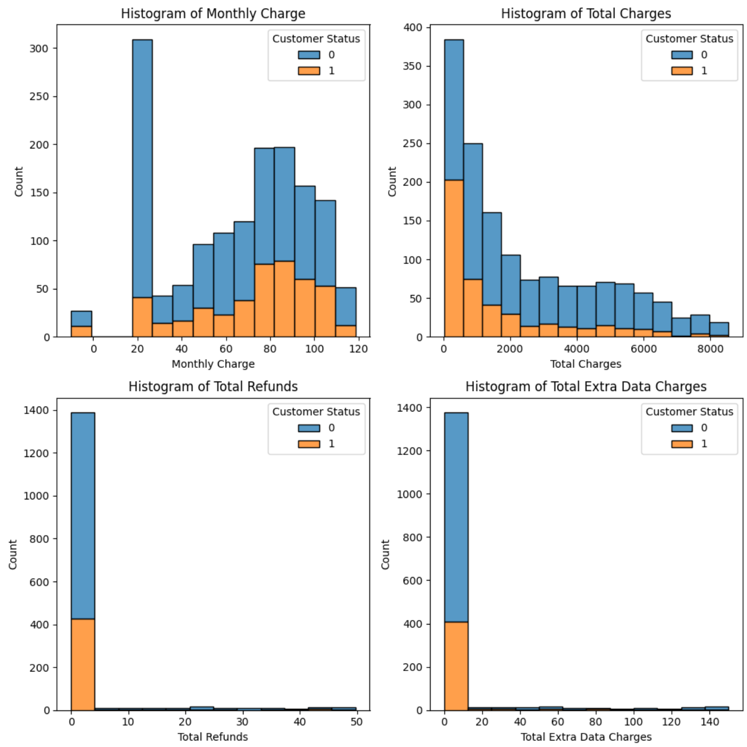

Figure 7: Histogram Plots for Monthly Charges, Total Charges, Total Refunds and Total Extra Data Charges

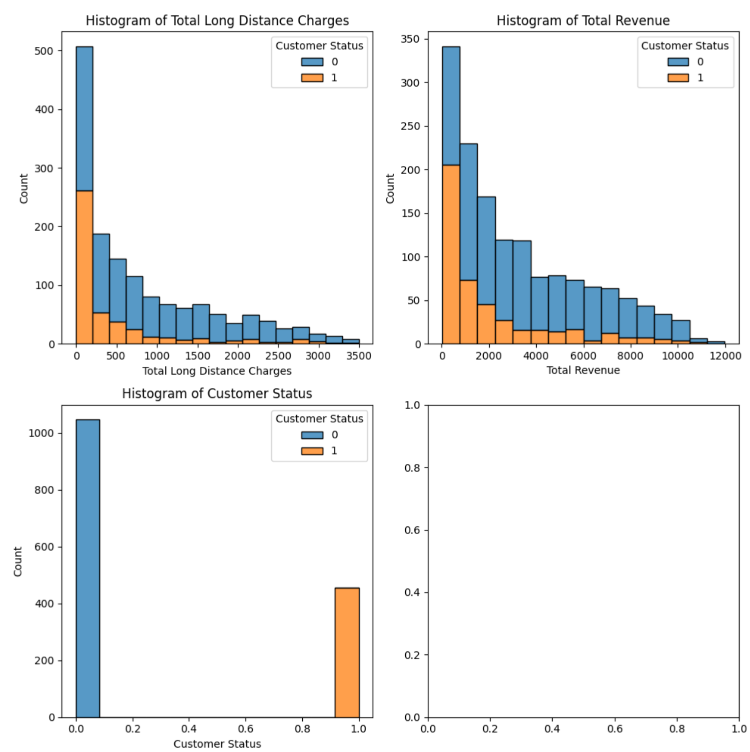

Figure 8: Histogram Plots for Total Long Distance Charges, Total Revenue and Customer Status

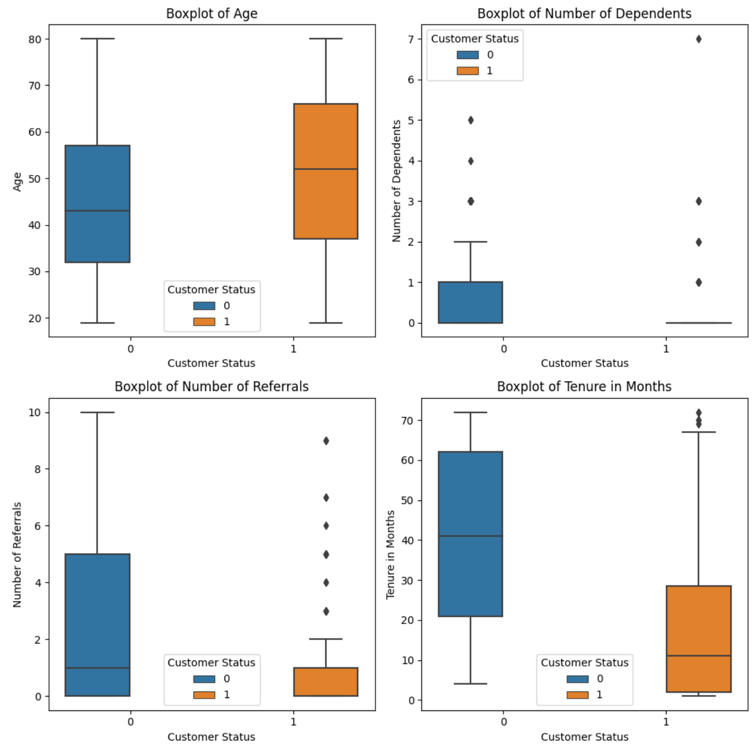

Figure 9: Boxplots for Age, Number of Dependents, Number of Referrals, and Tenure in Months

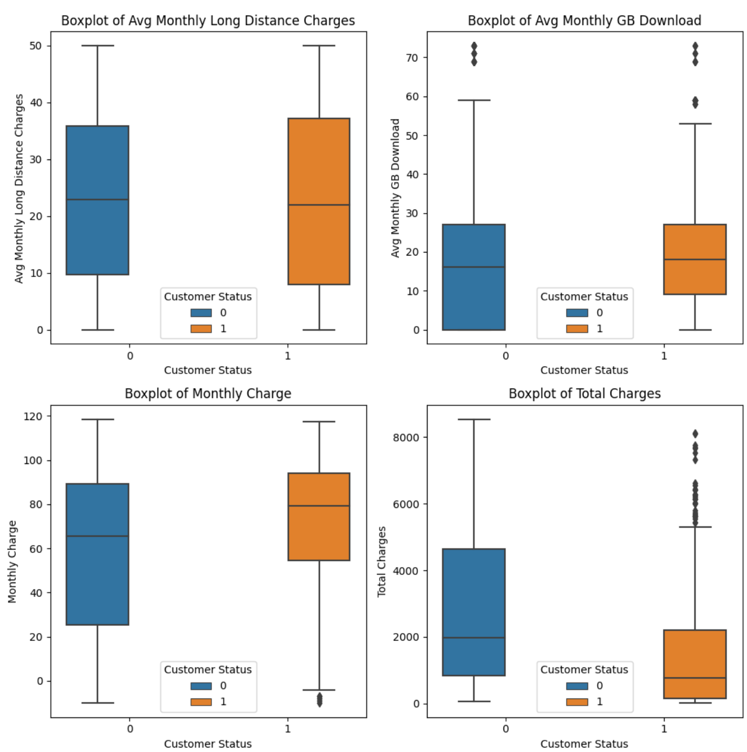

Figure 10: Boxplots for Avg Monthly Long Distance Charges, Avg Monthly GB Download, Monthly Charges and Total Charges

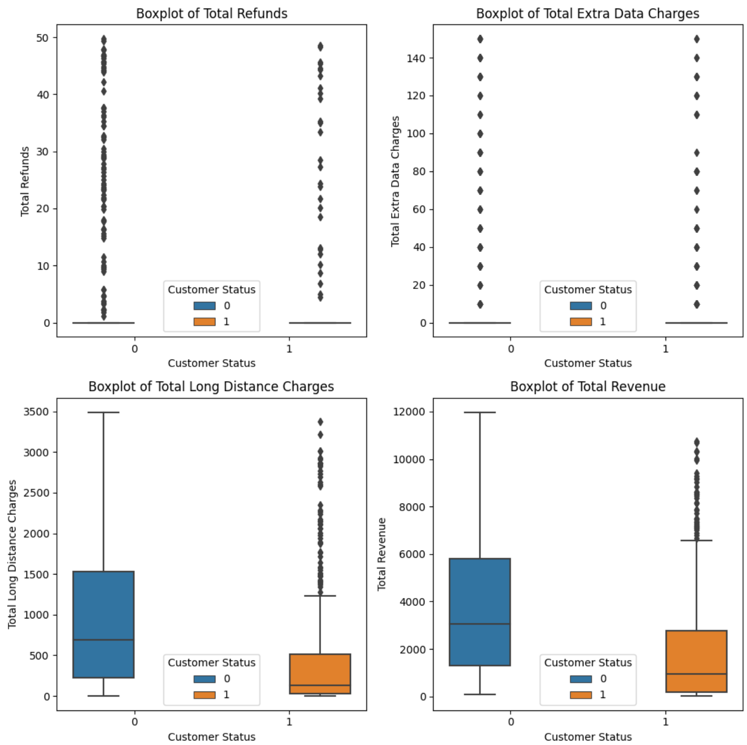

Figure 11: Boxplots for Total Refunds, Total Extra Data Charges, Total Long Distance Charges and Total Revenue

B Analyses

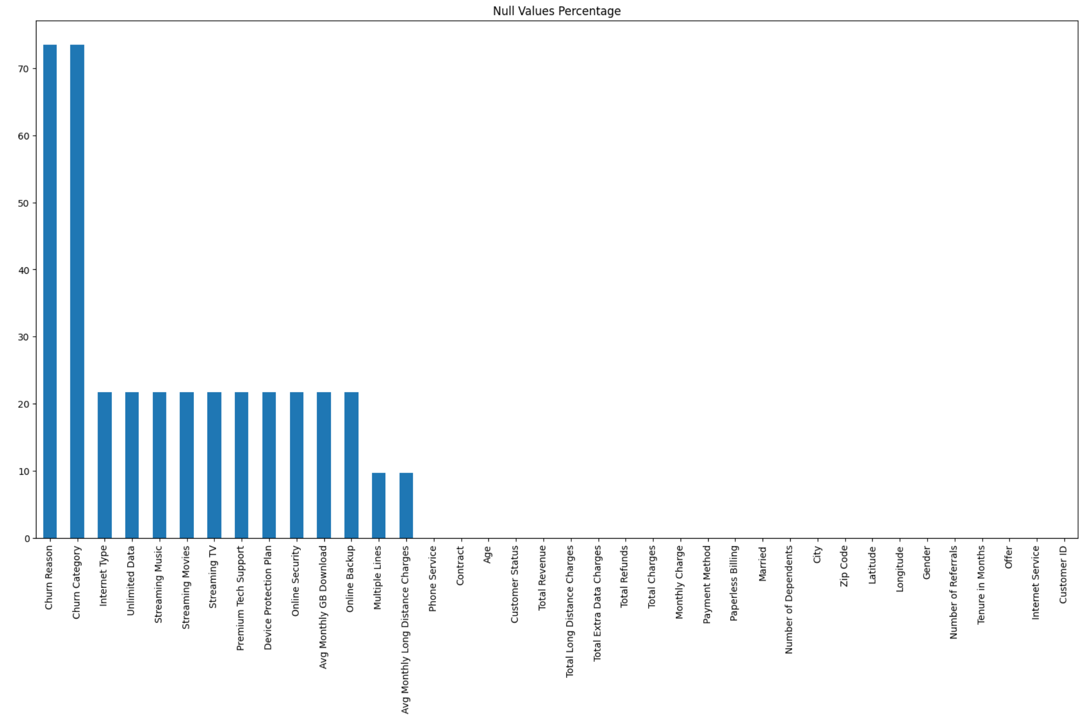

Figure 12: Percentage of Null Values Before Imputation

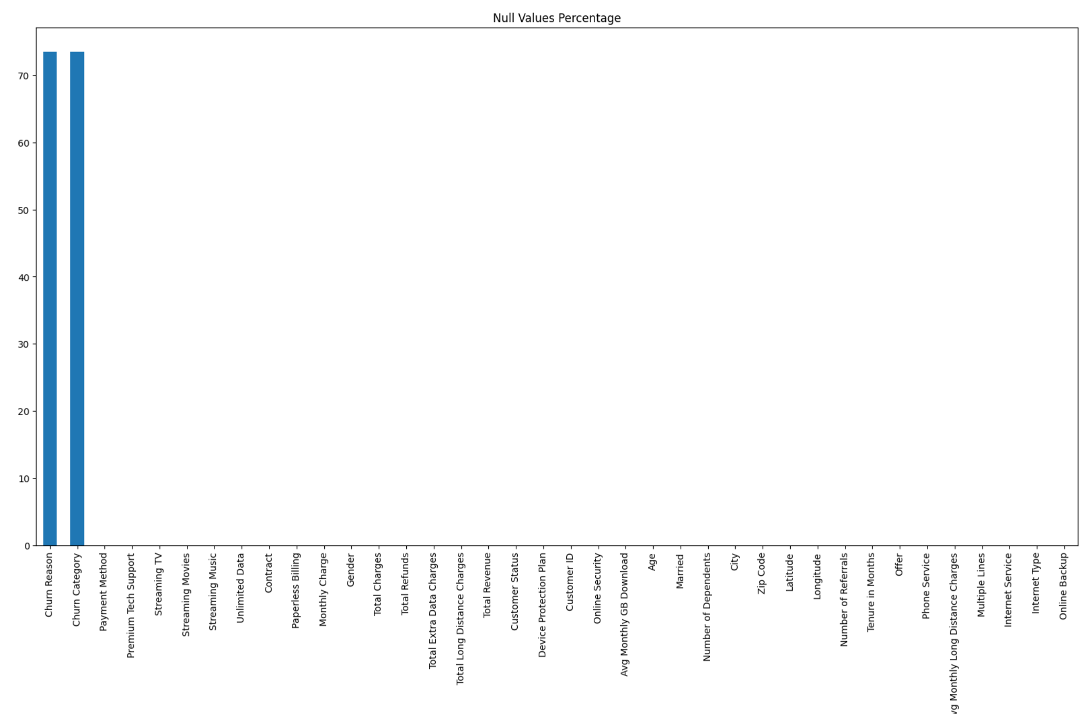

Figure 13: Percentage of Null Values After Imputation

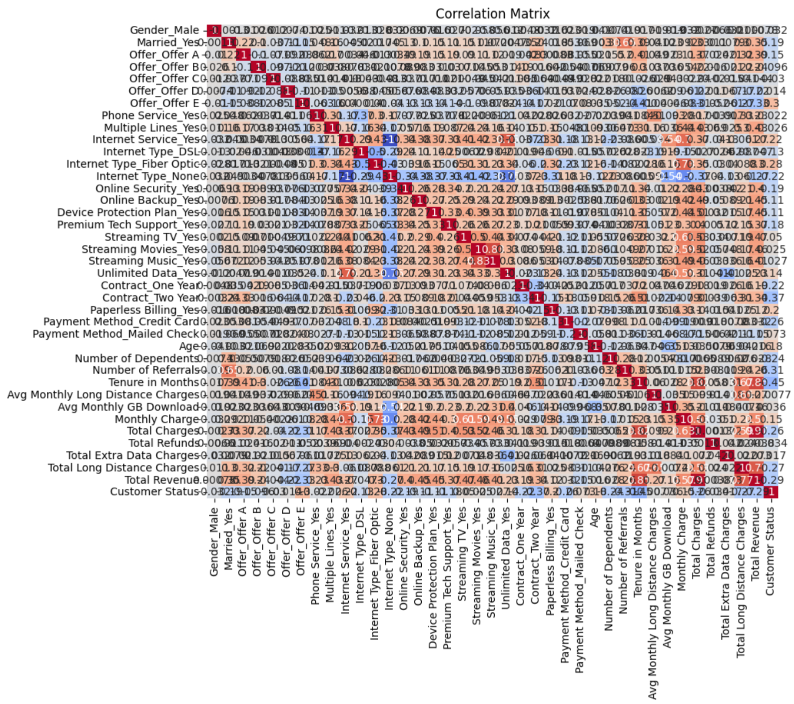

Figure 14: Correlation Matrix before Reducing Variables

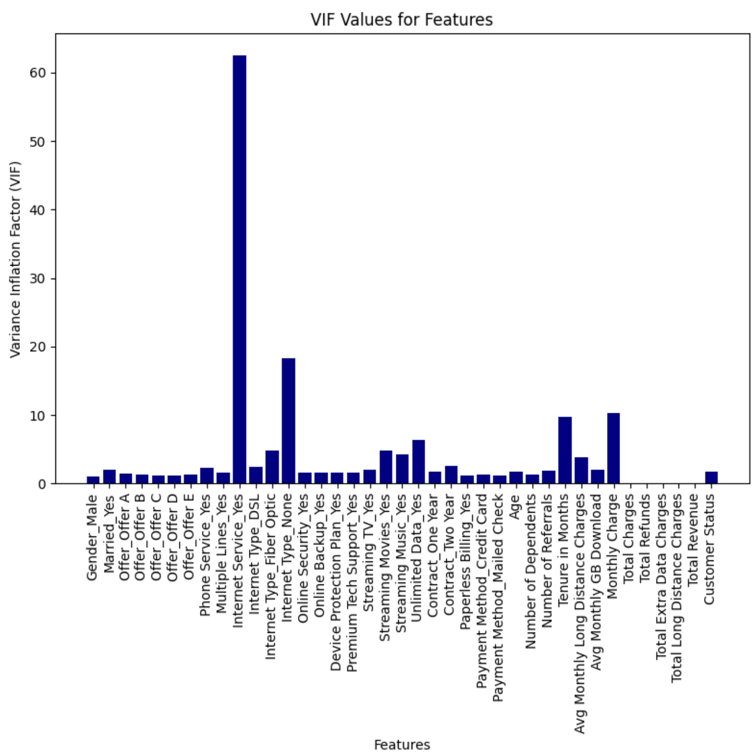

Figure 15: VIF Graph before Reducing Variables

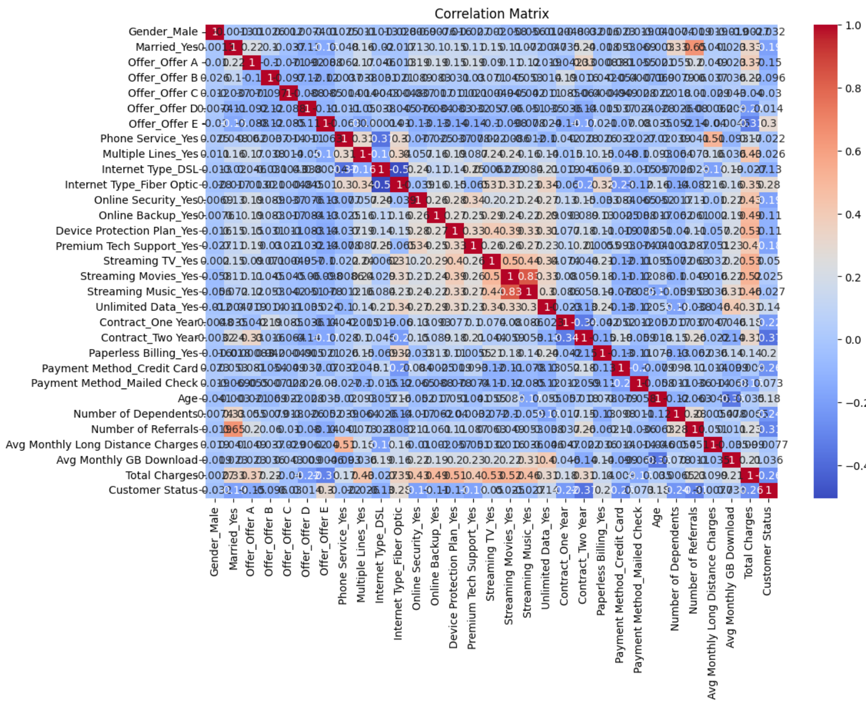

Figure 16: Correlation Matrix after Reducing Variables

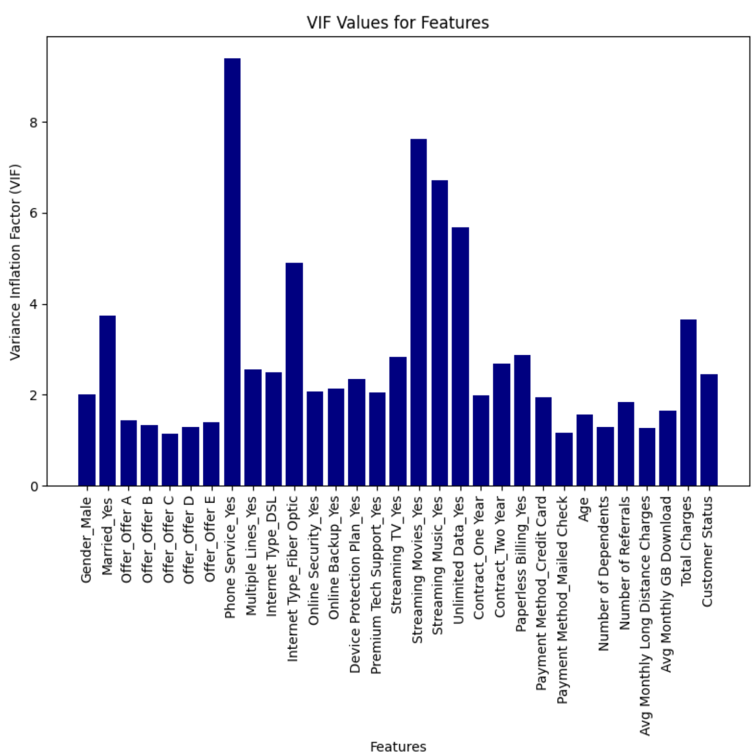

Figure 17: VIF Graph after Reducing Variables

Table 7: VIF values

feature	VIF	feature	**VIF**
Gender_Male	1.016237	Unlimited Data_Yes	6.366012
Married_Yes	2.033756	Contract_One Year	1.778808
Offer_Offer A	1.393786	Contract_Two Year	2.599480
Offer_Offer B	1.273029	Paperless Billing_Yes	1.219474
Offer_Offer C	1.085974	Payment Method_Credit Card	1.234493
Offer_Offer D	1.187615	Payment Method_Mailed Check	1.140997
Offer_Offer E	1.353305	Age	1.692528
Phone Service_Yes	2.233587	Number of Dependents	1.310725
Multiple Lines_Yes	1.514008	Number of Referrals	1.885556
Internet Service_Yes	62.569558	Tenure in Months	9.755316
Internet Type_DSL	2.385327	Avg Monthly Long Distance Charges	3.786101
Internet Type_Fiber Optic	4.835188	Avg Monthly GB Download	2.003158
Internet Type_None	18.334054	Monthly Charge	10.340883
Online Security_Yes	1.587076	Total Charges	inf
Online Backup_Yes	1.553148	Total Refunds	inf
Device Protection Plan_Yes	1.596247	Total Extra Data Charges	inf
Premium Tech Support_Yes	1.599662	Total Long Distance Charges	inf
Streaming TV_Yes	2.047351	Total Revenue	inf
Streaming Movies_Yes	4.874865	Customer Status	1.759383
Streaming Music_Yes	4.261304		

Table 8: Lasso Regression Coefficients

Feature	**Coefficient**
Gender_Male	-0.004998
Married_Yes	0.303740
Offer_Offer C	-0.240627
Offer_Offer D	-0.327839
Offer_Offer E	0.970290
Phone Service_Yes	-0.236950
Internet Type_DSL	-0.279559
Internet Type_Fiber Optic	0.731178
Online Security_Yes	-0.315313
Streaming TV_Yes	0.359674
Streaming Movies_Yes	0.278684
Contract_One Year	-1.400671
Contract_Two Year	-2.493032
Paperless Billing_Yes	0.394040
Payment Method_Credit Card	-0.822679
Age	0.384301
Number of Dependents	-0.504016
Number of Referrals	-0.837963
Avg Monthly Long Distance Charges	0.067116
Avg Monthly GB Download	0.156237
Total Charges	-0.419762

Table 9: Logistic Regression Model with Lasso Variables

Results: Logit				
Model:	Logit		Method:	MLE
Dependent Variable:	Customer Status		Pseudo R-squared:	0.444
No. Observations:	900		AIC:	644.6780
Df Model:	15		BIC:	721.5163
Df Residuals:	884		Log-Likelihood:	-306.34
Converged:	1.0000		LL-Null:	-551.46
No. Iterations:	8.0000		LLR p-value:	6.5308e-95
	Coef.	Std.Err.	z	P>—z—
const	-1.7485	0.3768	-4.6409	0.0000
Contract_Two Year	-2.9172	0.4481	-6.5107	0.0000
Contract_One Year	-1.5749	0.2927	-5.3813	0.0000
Offer_Offer E	1.3165	0.3591	3.6663	0.0002
Number of Referrals	-1.1554	0.2324	-4.9723	0.0000
Payment Method_Credit Card	-0.9116	0.2296	-3.9708	0.0001
Internet Type_Fiber Optic	0.9641	0.2962	3.2551	0.0011
Number of Dependents	-0.6459	0.1702	-3.7951	0.0001
Total Charges	-0.5431	0.1772	-3.0645	0.0022
Paperless Billing_Yes	0.5943	0.2292	2.5932	0.0095
Age	0.3274	0.1037	3.1582	0.0016
Streaming TV_Yes	0.7183	0.2469	2.9092	0.0036
Offer_Offer D	-0.5996	0.3206	-1.8702	0.0615
Online Security_Yes	-0.3397	0.2407	-1.4113	0.1581
Married_Yes	0.7848	0.2886	2.7196	0.0065
Internet Type_DSL	-0.1997	0.3055	-0.6536	0.5134

Table 10: Logistic Regression Model with Stepwise Variables

Results: Logit				
Model:	Logit		Method:	MLE
Dependent Variable:	Customer Status		Pseudo R-squared:	0.400
No. Observations:	900		AIC:	685.2256
Df Model:	11		BIC:	742.8544
Df Residuals:	888		Log-Likelihood:	-330.61
Converged:	1.0000		LL-Null:	-551.46
No. Iterations:	8.0000		LLR p-value:	8.4127e-88
	Coef.	Std.Err.	z	P>—z—
const	-1.7888	0.2616	-6.8375	0.0000
Married_Yes	0.4052	0.2588	1.5659	0.1174
Offer_Offer A	0.7317	0.6500	1.1257	0.2603
Offer_Offer B	-0.1819	0.3535	-0.5146	0.6068
Offer_Offer E	1.5999	0.3277	4.8818	0.0000
Internet Type_Fiber Optic	0.9414	0.2146	4.3863	0.0000
Streaming Movies_Yes	0.5181	0.2160	2.3993	0.0164
Contract_One Year	-1.8575	0.2733	-6.7964	0.0000
Contract_Two Year	-3.6186	0.4772	-7.5834	0.0000
Paperless Billing_Yes	0.6755	0.2192	3.0824	0.0021
Age	0.4019	0.1001	4.0157	0.0001
Number of Referrals	-1.2119	0.2263	-5.3551	0.0000

Table 11: Logistic Regression Model with All Variables

Results: Logit				
Model:	Logit		Method:	MLE
Dependent Variable:	Customer Status		Pseudo R-squared:	0.4591
No. Observations:	900		Log-Likelihood:	-298.30
Df Model:	30		LL-Null:	-551.46
Df Residuals:	869		LLR p-value:	6.098e-88
	Coef.	Std.Err.	z	P>—z—
const	-1.5736	0.584	-2.694	0.007
Gender_Male	-0.0314	0.212	-0.148	0.883
Married_Yes	0.7156	0.294	2.430	0.015
Offer_Offer A	1.8211	0.747	2.436	0.015
Offer_Offer B	0.1929	0.405	0.477	0.634
Offer_Offer C	-0.5724	0.431	-1.329	0.184
Offer_Offer D	-0.7030	0.335	-2.098	0.036
Offer_Offer E	1.1737	0.371	3.166	0.002
Phone Service_Yes	-0.2753	0.423	-0.650	0.516
Multiple Lines_Yes	-0.0749	0.262	-0.285	0.775
Internet Type_DSL	-0.3365	0.342	-0.984	0.325
Internet Type_Fiber Optic	0.9379	0.353	2.660	0.008
Online Security_Yes	-0.4023	0.248	-1.620	0.105
Online Backup_Yes	-0.0737	0.259	-0.285	0.776
Device Protection Plan_Yes	0.0333	0.254	0.131	0.896
Premium Tech Support_Yes	-0.0624	0.266	-0.235	0.814
Streaming TV_Yes	0.6164	0.266	2.317	0.021
Streaming Movies_Yes	0.5782	0.424	1.364	0.172
Streaming Music_Yes	-0.1942	0.419	-0.463	0.643
Unlimited Data_Yes	0.1392	0.278	0.501	0.617
Contract_One Year	-1.5816	0.302	-5.233	0.000
Contract_Two Year	-3.2416	0.514	-6.307	0.000
Paperless Billing_Yes	0.5378	0.241	2.234	0.026
Payment Method_Credit Card	-0.8560	0.242	-3.542	0.000
Payment Method_Mailed Check	0.2551	0.411	0.620	0.535
Age	0.4113	0.133	3.097	0.002
Number of Dependents	-0.6349	0.175	-3.633	0.000
Number of Referrals	-1.1634	0.234	-4.962	0.000
Avg Monthly Long Distance Charges	0.0947	0.123	0.771	0.441
Avg Monthly GB Download	0.1992	0.131	1.521	0.128
Total Charges	-0.6926	0.238	-2.914	0.004

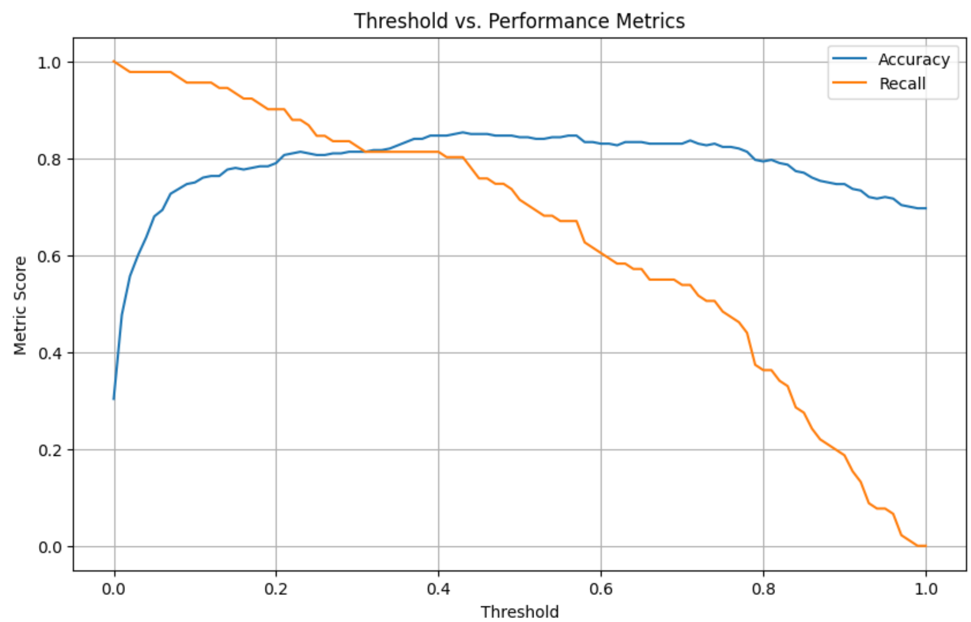

Figure 18: Threshold vs. Accuracy and Recall

C Python Code

Note: to see our code full with comments and graphs included, see the jupyter notebook file.

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import chi2

import statsmodels.api as sm
from statsmodels.stats.outliers_influence import variance_inflation_factor

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score,
roc_auc_score
from sklearn.linear_model import LogisticRegressionCV
from sklearn.feature_selection import SequentialFeatureSelector
```

Listing 1: Importing Libraries

```python
data = pd.read_csv('telecom_customer_churn.csv')

# imputation
missing_values = data.isnull().sum()

data['Avg Monthly Long Distance Charges'].fillna(0, inplace=True)
data['Multiple Lines'].fillna('No', inplace=True)
data['Offer'].fillna('None', inplace=True)
data['Internet Type'].fillna('None', inplace=True)
data['Avg Monthly GB Download'].fillna(0, inplace=True)

columns_to_fill = ['Online Security', 'Online Backup', 'Device Protection Plan',
                   'Premium Tech Support', 'Streaming TV', 'Streaming Movies',
                   'Streaming Music', 'Unlimited Data']
```

```python

for col in columns_to_fill:
    data[col].fillna('No', inplace=True)

data_no_join = data[data['Customer Status'] != 'join']
data_no_join = data[data['Customer Status'] != 'Joined']

columns_to_drop = ['Customer ID','City', 'Zip Code', 'Latitude', 'Longitude', 'Churn Category', 'Churn Reason']
data_no_location = data_no_join.drop(columns=columns_to_drop)

# outliers
numerical_columns_full = data_no_location.select_dtypes(include=['int64', 'float64']).columns

data_full_adjusted_outliers = data_no_location.copy()

excluded_columns = ['Number of Dependents', 'Number of Referrals', 'Total Refunds', 'Total Extra Data Charges']
iqr_multiplier_full = 2  # Using a multiplier of 2


for col in numerical_columns_full:
    if col not in excluded_columns:
        Q1 = data_full_adjusted_outliers[col].quantile(0.25)
        Q3 = data_full_adjusted_outliers[col].quantile(0.75)
        IQR = Q3 - Q1
        lower_bound = Q1 - iqr_multiplier_full * IQR
        upper_bound = Q3 + iqr_multiplier_full * IQR

        # Filtering the data
        filter_condition = (data_full_adjusted_outliers[col] >= lower_bound) & (data_full_adjusted_outliers[col] <= upper_bound)
        data_full_adjusted_outliers = data_full_adjusted_outliers[filter_condition]

# sample data
sample_data = data_full_adjusted_outliers.sample(n=1500, random_state=42)
sample_data['Customer Status'] = sample_data['Customer Status'].replace({'Stayed': 0, 'Churned': 1})
```

Listing 2: Data Preprocessing

```python
for x in sample_data.columns:
    sns.histplot(x=x,data=sample_data,hue='Customer Status', multiple = "stack")
    plt.title(f'Histogram of {x}')
    plt.show()

cols = list(sample_data.select_dtypes(include=['object']).columns)
df = sample_data.drop(cols, axis=1)
for x in df.columns:
    sns.boxplot(y=x,x='Customer Status',data=df,hue='Customer Status')
    plt.title(f'Histogram of {x}')
    plt.show()

# encode categorical variables
categorical_columns = sample_data.select_dtypes(include=['object']).columns

preprocessor = ColumnTransformer(
    transformers=[
        ('cat', OneHotEncoder(drop='first'), categorical_columns)
    ], remainder='passthrough')

sample_data_encoded = preprocessor.fit_transform(sample_data)

new_feature_names = preprocessor.named_transformers_['cat'].get_feature_names_out(categorical_columns)
new_feature_names = list(new_feature_names) + [col for col in sample_data.columns if col not in categorical_columns]
```

```python

    sample_data_encoded_df = pd.DataFrame(sample_data_encoded, columns=new_feature_names)

    # scale numerical variables
    numerical_columns = sample_data.select_dtypes(include=['int64', 'float64']).columns.drop
    ('Customer Status')
    scaler = StandardScaler()
    sample_data_encoded_df[numerical_columns] = scaler.fit_transform(sample_data[
    numerical_columns])

    # VIF calculation
    numeric_data = sample_data_encoded_df.select_dtypes(include=[np.number])

    vif_data = pd.DataFrame()
    vif_data["feature"] = numeric_data.columns
    vif_data["VIF"] = [variance_inflation_factor(numeric_data.values, i) for i in range(len(
    numeric_data.columns))]

    columns_to_remove = ['Monthly Charge', 'Tenure in Months', 'Total Refunds', 'Total Extra
     Data Charges', 'Total Long Distance Charges', 'Total Revenue', 'Internet Service_Yes',
    'Internet Type_None']
    sample_data_reduced = sample_data_encoded_df.drop(columns=columns_to_remove)
    numeric_data_reduced = sample_data_reduced.select_dtypes(include=[np.number])

    numeric_data_reduced = sample_data_reduced.select_dtypes(include=[np.number])
    vif_data_reduced = pd.DataFrame()
    vif_data_reduced["feature"] = numeric_data_reduced.columns
    vif_data_reduced["VIF"] = [variance_inflation_factor(numeric_data_reduced.values, i) for
     i in range(len(numeric_data_reduced.columns))]
```

Listing 3: Data Preprocessing

```python

    # full model
    X = sample_data_reduced.drop('Customer Status', axis=1)
    y = sample_data_reduced['Customer Status']

    X_train, X_temp, y_train, y_temp = train_test_split(X, y, test_size=0.4, stratify=y,
    random_state=42)
    X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp, test_size=0.5, stratify=
    y_temp, random_state=42)

    X_train_sm = sm.add_constant(X_train)
    logit_model = sm.Logit(y_train, X_train_sm)
    result = logit_model.fit()
    print(result.summary())

    y_train_pred_sm = result.predict(X_train_sm)
    y_train_pred_binary = [1 if x > 0.5 else 0 for x in y_train_pred_sm]

    accuracy = accuracy_score(y_train, y_train_pred_binary)
    precision = precision_score(y_train, y_train_pred_binary)
    recall = recall_score(y_train, y_train_pred_binary)
    f1 = f1_score(y_train, y_train_pred_binary)
    roc_auc = roc_auc_score(y_train, y_train_pred_sm)
```

Listing 4: Model Fitting

```python
    # lasso
    lasso_log_reg = LogisticRegressionCV(Cs=10, cv=5, penalty='l1', solver='liblinear',
    random_state=42, max_iter=1000)
    lasso_log_reg.fit(X_train, y_train)

    best_alpha = 1 / lasso_log_reg.C_[0]

    lasso_log_reg_optimized = LogisticRegression(penalty='l1', C=1/best_alpha, solver='
    liblinear', random_state=42, max_iter=1000)
    lasso_log_reg_optimized.fit(X_train, y_train)
```

```python

    coefficients = lasso_log_reg_optimized.coef_[0]
    features_coefficients = pd.DataFrame({'Feature': X_train.columns, 'Coefficient': coefficients})
    non_zero_features = features_coefficients[features_coefficients['Coefficient'] != 0]

    top_n_features = non_zero_features.assign(Abs_Coefficient=non_zero_features['Coefficient'].abs())
    top_n_features_sorted = top_n_features.sort_values(by='Abs_Coefficient', ascending=False).head(15)

    summary = result.summary2().tables[1]
    significant_features = summary[summary['P>|z|'] < 0.05]['Coef.']

    selected_features = top_n_features_sorted['Feature']
    X_train_selected = X_train[selected_features]

    X_train_selected_with_const = sm.add_constant(X_train_selected)
    X_test_selected_with_const = sm.add_constant(X_test[selected_features], has_constant='add')

    model = sm.Logit(y_train, X_train_selected_with_const)
    result = model.fit()

    print(result.summary2())# Fit the logistic regression model

    significant_predictors = significant_features.index.tolist()
    significant_predictors = [predictor for predictor in significant_predictors if predictor != 'const']

    X_train_sig = X_train[significant_predictors]
    X_test_sig = X_test[significant_predictors]
    X_train_sig_with_const = sm.add_constant(X_train_sig)
    X_test_sig_with_const = sm.add_constant(X_test_sig)

    model = sm.Logit(y_train, X_train_sig_with_const)
    result_lasso = model.fit()
    print(result_lasso.summary2())

    predictions_prob_lasso = result_lasso.predict(X_test_sig_with_const)
    predictions_lasso = np.where(predictions_prob_lasso > 0.5, 1, 0)  # Convert probabilities to 0/1

    accuracy_lasso = np.mean(predictions_lasso == y_test)
    recall_lasso = recall_score(y_test, predictions_lasso)
    auc_roc_lasso = roc_auc_score(y_test, predictions_prob_lasso)

    # subset vs full
    X_train_full = sm.add_constant(X_train)

    logit_model_full = sm.Logit(y_train, X_train_full)
    result_full = logit_model_full.fit()

    significant_predictors = significant_features.index.tolist()
    significant_predictors = [predictor for predictor in significant_predictors if predictor != 'const']

    X_train_sig = X_train[significant_predictors]
    X_test_sig = X_test[significant_predictors]

    X_train_sig_with_const = sm.add_constant(X_train_sig)
    X_test_sig_with_const = sm.add_constant(X_test_sig)

    model_reduced = sm.Logit(y_train, X_train_sig_with_const)
    result_reduced = model_reduced.fit()

    lr_statistic = -2 * (result_reduced.llf - result_full.llf)

```

```python
    df = (len(result_full.params) - len(result_reduced.params))

    p_value_lr_test = 1 - chi2.cdf(lr_statistic, df)

    print(f"Likelihood Ratio Test Statistic: {lr_statistic:.2f}")
    print(f"Degrees of Freedom: {df}")
    print(f"P-Value (LR Test): {p_value_lr_test:.4f}")

    # goodness of fit
    log_likelihood_train_updated = np.sum(np.log(result.predict(X_train_sig_with_const) ** y_train *
                                  (1 - result.predict(X_train_sig_with_const)) ** (1 - y_train)))
    log_likelihood_test_updated = np.sum(np.log(result.predict(X_test_sig_with_const) ** y_test *
                                  (1 - result.predict(X_test_sig_with_const)) ** (1 - y_test)))

    deviance_test_updated = -2 * (log_likelihood_test_updated - log_likelihood_train_updated)

    df_deviance_test_updated = X_test_sig_with_const.shape[0] - (X_test_sig_with_const.shape[1])

    p_value_deviance_test_updated = 1 - chi2.cdf(deviance_test_updated, df_deviance_test_updated)
```

Listing 5: Variable Selection with Lasso

```python
    # forward stepwise regression
    log_reg = LogisticRegression()
    sfs = SequentialFeatureSelector(log_reg, direction='forward', cv=5)
    sfs.fit(X_train, y_train)
    forward_selected_features = X_train.columns[sfs.get_support()]
    selected_features = []

    for feature in forward_selected_features:
        model = sm.Logit(y_train, sm.add_constant(X_train[feature]))
        result = model.fit()
        if result.pvalues[1] < 0.05:
            selected_features.append(feature)

    forward_selected_features_ = selected_features

    # backward
    sbs = SequentialFeatureSelector(log_reg, direction='backward', cv=5)
    sbs.fit(X_train, y_train)
    backward_selected_features = X_train.columns[sfs.get_support()]
    selected_features = []

    for feature in backward_selected_features:
        model = sm.Logit(y_train, sm.add_constant(X_train[feature]))
        result = model.fit()
        if result.pvalues[1] < 0.05:
            selected_features.append(feature)

    backward_selected_features_ = selected_features

    # model fitting
    X_train_backward_with_const = sm.add_constant(X_train[backward_selected_features_])
    X_test_backward_with_const = sm.add_constant(X_test[backward_selected_features_])

    model_backward = sm.Logit(y_train, X_train_backward_with_const)
    result_backward = model_backward.fit()

    print(result_backward.summary2())

    predictions_prob_backward = result_backward.predict(X_test_backward_with_const)
```

```python
predictions_backward = np.where(predictions_prob_backward > 0.5, 1, 0)  # Convert probabilities to 0/1

accuracy_backward = np.mean(predictions_backward == y_test)
recall_backward = recall_score(y_test, predictions_backward)
auc_roc_backward = roc_auc_score(y_test, predictions_prob_backward)

# testing subset vs full
X_train_full = sm.add_constant(X_train)
logit_model_full = sm.Logit(y_train, X_train_full)
result_full = logit_model_full.fit()
significant_predictors = backward_selected_features_

X_train_sig = X_train[significant_predictors]
X_test_sig = X_test[significant_predictors]
X_train_sig_with_const = sm.add_constant(X_train_sig)
X_test_sig_with_const = sm.add_constant(X_test_sig)

model_reduced = sm.Logit(y_train, X_train_sig_with_const)
result_reduced = model_reduced.fit()

lr_statistic = -2 * (result_reduced.llf - result_full.llf)
df = (len(result_full.params) - len(result_reduced.params))
p_value_lr_test = 1 - chi2.cdf(lr_statistic, df)

print(f"Likelihood Ratio Test Statistic: {lr_statistic:.2f}")
print(f"Degrees of Freedom: {df}")
print(f"P-Value (LR Test): {p_value_lr_test:.4f}")

# goodness of fit
log_likelihood_train_backward = np.sum(np.log(result_backward.predict(X_train_backward_with_const) ** y_train *
                                   (1 - result_backward.predict(X_train_backward_with_const)) ** (1 - y_train)))
log_likelihood_test_backward = np.sum(np.log(result_backward.predict(X_test_backward_with_const) ** y_test *
                                   (1 - result_backward.predict(X_test_backward_with_const)) ** (1 - y_test)))

deviance_test_backward = -2 * log_likelihood_test_backward
df_deviance_test_backward = X_test_backward_with_const.shape[0] - (X_test_backward_with_const.shape[1])
p_value_deviance_test_backward = 1 - chi2.cdf(deviance_test_backward, df_deviance_test_backward)
```

Listing 6: Variable Selection with Stepwise Regression

```python

model_metrics = [
    {"Model": "Logistic Regression Model(lasso)", "Accuracy": accuracy_lasso, "Recall": recall_lasso, "AUC-ROC": auc_roc_lasso},
    {"Model": "Logistic Regression Model(stepwise)", "Accuracy": accuracy_backward, "Recall": recall_backward, "AUC-ROC": auc_roc_backward},
]
metrics_df = pd.DataFrame(model_metrics)
metrics_df

print(result_lasso.summary2())
```

Listing 7: Comparing Models

```python
# tuning
thresholds = np.arange(0, 1.01, 0.01)
accuracy_scores = []
recall_scores = []

for threshold in thresholds:
    binary_predictions = np.where(predictions_prob_lasso > threshold, 1, 0)
```

```
        accuracy = np.mean(binary_predictions == y_test)
        recall = recall_score(y_test, binary_predictions)
        accuracy_scores.append(accuracy)
        recall_scores.append(recall)

    best_recall_threshold = thresholds[np.argmax(recall_scores)]
    max_recall = max(recall_scores)
    best_accuracy_threshold = thresholds[np.argmax(accuracy_scores)]
    max_accuracy = max(accuracy_scores)

    print(f"Best Accuracy: {max_accuracy:.2f} at Threshold: {best_accuracy_threshold:.2f}")

    idx = np.argwhere(np.diff(np.sign(np.array(accuracy_scores) - np.array(recall_scores))))
    .flatten()
    print(f"Best threshold: {thresholds[idx]} with an accuracy of {np.array(accuracy_scores)
    [idx]} and recall of {np.array(recall_scores)[idx]}")

    # refitting model with threshold
    significant_predictors = significant_features.index.tolist()
    significant_predictors = [predictor for predictor in significant_predictors if predictor
     != 'const']

    X_train_sig = X_train[significant_predictors]
    X_test_sig = X_test[significant_predictors]
    X_train_sig_with_const = sm.add_constant(X_train_sig)
    X_test_sig_with_const = sm.add_constant(X_test_sig)

    model = sm.Logit(y_train, X_train_sig_with_const)
    result_lasso = model.fit()
    print(result_lasso.summary2())

    predictions_prob_lasso = result_lasso.predict(X_test_sig_with_const)
    predictions_lasso = np.where(predictions_prob_lasso > 0.43, 1, 0)

    accuracy_lasso = np.mean(predictions_lasso == y_test)
    recall_lasso = recall_score(y_test, predictions_lasso)
    auc_roc_lasso = roc_auc_score(y_test, predictions_prob_lasso)
```

Listing 8: Tuning Threshold

```
    # model validation
    significant_predictors = significant_features.index.tolist()
    significant_predictors = [predictor for predictor in significant_predictors if predictor
     != 'const']

    X_val_sig = X_val[significant_predictors]
    X_val_sig_with_const = sm.add_constant(X_val_sig)

    predictions_prob_val = result_lasso.predict(X_val_sig_with_const)
    predictions_val = np.where(predictions_prob_val > 0.43, 1, 0)  # Convert probabilities
    to 0/1

    accuracy_val = np.mean(predictions_val == y_val)
    recall_val = recall_score(y_val, predictions_val)
    auc_roc_val = roc_auc_score(y_val, predictions_prob_val)
```

Listing 9: Model Validation

References

[1] Merriam-Webster, "Churn," in *Merriam-Webster.com dictionary*. [Online]. Available: https://www.merriam-webster.com/dictionary/churn

[2] S. L. Zhuang, "Telecom customer churn prediction," Tech. Rep., 2022, https://www.kaggle.com/datasets/shilongzhuang/telecom-customer-churn-by-maven-analytics.

[3] Mar 2022. [Online]. Available: https://www.datacamp.com/tutorial/tutorial-lasso-ridge-regression#