SyriaTel Churn Modeling: Predicting Customer Retention.

Business Understanding

Business overview

SyriaTel is a mobile communication service company headquartered in Damascus, Syria. It is the best performing mobile operator and operates the 2G and 3G network in the country. It has over two hundred international partners spread across 121 countries (Tracxn, 2024). The company's mobile services include data, news, and voice, roaming and messaging. Like many other tele-communication firms, SyriaTel deals with the challenge of client attrition. It is important to identify potentially dissatisfied customers before switching to other service providers. This would enable the company to raise their bottom line and retain more of its customers. According to Simplilearn (2023), the evaluation of customer's loss rate is referred to as churn analysis. It allows for the company to know the rate at which its customers opt out from its services.

Recruiting new customers is frequently more expensive than keeping existing ones, churn can be pricey. In particular, SyriaTel has seen swings in customer retention rates, raising questions about how to effectively hold onto valuable clients. By spotting churn early on, the business can take proactive steps to keep these clients, this retains the long-term profitability. Due to the potential impact on profitability, churn analysis becomes vital for telecom firms such as SyriaTel. By precisely determining which customers are most likely to leave, SyriaTel can make efficient use of its resources to keep consumers and increase customer retention. The organization can concentrate on analyzing customer data and try deriving insights that lead to customer churning. SyriaTel may use machine learning algorithms to make data-driven decisions and create successful retention strategies by gaining insightful knowledge about customer behavior and churn-influencing factors. Predicting customer attrition is the goal in order to help SyriaTel actively keep high-risk clients.

Problem Statement

SyriaTel Company lacks a reliable means to identify which customers are most likely to churn. The lack of business strategy to identify this group of customers raises the turnover rates and expenses related to customer churning. The specific questions to be addressed are;

i. What are the major factors that influence the customer churn?
ii. Can we identify and predict customer churn patterns?
iii. What model is the most efficient in predicting customer churn?

Project Objectives

Primary Objective

To develop a classification model that accurately predicts customer churn based on the provided data, enabling SyriaTel to implement proactive and effective customer retention strategies.

Secondary Objective

Identify the factors that most significantly contribute to customer churn.

Identify and analyze patterns related to customer churn behavior.

Approach Methodology

An approach that best addresses this problem is to perform a thorough exploration on data to understand the relationship between features. The first step is to determine the relationship between factors influencing customer churn. To assess the association between these variables, we conduct a correlation analysis, and determine the strength of correlations. To visually identify these factors, we display a heat map for the correlation values, which helps in identifying the most influential factors to customer churn. Correlation analysis also helps in identifying multicolinearity, a situation where we can perfectly predict another feature. This is not desirable because it leads to dummy variable trap.

From the correlation analysis, we obtain the most influential factors to customer churn. Other important analysis in addressing the questions, univariate, bivariate and bivariate analysis. The univariate, bivariate and multivariate analysis further reveals the patterns and movement of the features in relation to customer churn. These analysis forms the baseline for modelling various classification problems. Modelling comprise the creation of the prediction using the algorithms such as logistics regression, decision trees classifies and random forests. These models heavily relies on the decisions made in the data exploration stage but further alterations in the data are done such as transformations.

Metrics of Success

The accepted industrial standards; precision, recall, accuracy and f1-score in predicting customer churn for telecom companies varies across companies and depending on the business context (ResearchGate, 2020). However, the general metrics are defined for above average performing models. What we consider as acceptable performance for the churn project prediction model is:

Accuracy level > 70%. Measure the percentage of correctly predicting instances for both churners and non-churners. Higher accuracy is desirable and appealing but not so much informative especially for imbalanced datasets

Precision > 70%. This measures the proportion of positive predictions for the churners that are actually correct. Higher precision is indicative of the model's ability to predict churners without flagging too many non-churners. For telecom companies, precision is desirable where the company has limited resources and wants to avoid unnecessary costs on interventions. A precision above 70% is considered sufficient.

Recall >60%. This measures the percentage of actual churners that are correctly identified as churners. High recall implies that the model detects most of the churners and thus reduces the level of false negatives.

F1-score > 70%. This is indicative of the balance between precision and recall. It shows the middle ground between precision and recall. This metric is important when both false positives and false negatives are both costly to the company.

Data Understanding

Obtaining the data

We used the provided customer churn data set for SyriaTel Company. Specifically, we utilized the data from Kaggle.com. The data set was read using the Pandas library as it was in the .csv format. This allowed for further data inspection. By using the .info and .describe() methods we saw the data set constituted of 21 columns and 3333 rows and the data features were both categorical and numerical and with a mix of object, integers and bool data types. The description of these columns are;

Categorical Features:

State: The state where the customer resides.

International plan: Whether the customer has an international plan (Yes or No).

Phone number: The phone number of the customer.

Voice mail plan: Whether the customer has a voice mail plan (Yes or No).

Numeric Features:

Area code: The area code associated with the customer's phone number.

Account length: The number of days the customer has been an account holder.

Number vmail messages: The number of voice mail messages received by the customer.

Total day minutes: The total number of minutes the customer used during the day.

Total day calls: The total number of calls made by the customer during the day.

Total day charge: The total charges incurred by the customer for daytime usage.

Total eve minutes: The total number of minutes the customer used during the evening.

Total eve calls: The total number of calls made by the customer during the evening.

Total eve charge: The total charges incurred by the customer for evening usage.

Total night minutes: The total number of minutes the customer used during the night.

Total night calls: The total number of calls made by the customer during the night.

Total night charge: The total charges incurred by the customer for nighttime usage.

Total intl minutes: The total number of international minutes used by the customer.

Total intl calls: The total number of international calls made by the customer.

Total intl charge: The total charges incurred by the customer for international usage.

Customer service calls: The number of customer service calls made by the customer.

Data Preparation

Data cleaning.

We used various approaches to detect the undesirable data. The .isna(), and .duplicated() methods allowed us to detect the presence of null values and duplicates. The data set did not have any null nor duplicated values. The columns such as area codes were numerical values but at an ordinal level, these values stood as discrete categorical since the difference between these values were not meaningful. We thus changed this column to datatype object. The phone number column was not o so much importance, thus dropping would suffice.

We checked for outliers, the values that fall beyond the normal distribution, for each column. We identified the 25^{th} and the 75^{th} quartiles and with these values we computed the interquartile range. The interquartile range is the difference between the 25^{th} and 75^{th} quantiles. Outliers in predictive models can cause overfitting, reduced accuracy, and instability during training. Excluding outliers, which are data points beyond 3 standard deviations from the mean, improves accuracy and stable training. Eliminating outliers aligns with algorithms' assumptions and enhances model performance on unseen data, ensuring a normal distribution.

Exploratory Data Analysis

Exploratory Data Analysis (EDA) forms the foundational step in data analysis. We concentrate on exploring and understanding the dataset to uncover patterns, relationships, and insights before applying modeling or statistical techniques. Through visualizations such as scatter plots, histograms, and box plots, EDA helps reveal trends and distributions in the data, while correlation analysis identifies relationships between variables. We assess the impact of outliers using statistical methods, computation of the IQR from which we calculated the upper and lower bounds. By conducting EDA, we will be able to gather first insights and make informed decisions on further analysis. We perform various analyses such as, univariate, bivariate, and multivariate analysis.

i. Univariate analysis

Before continuing to predict customer churn, we can take a brief detour into analyzing the distribution of individual features. The characteristics of our data, the univariate analysis parameters and the representation of the descriptive information of features can be illustrated by presenting feature distributions (Tessler, 2023). We thus write a function that takes in the data and plots the distribution. We call the function and pass in our data frame as the argument. We pass in both the categorical features and numerical features and display their distributions. We have three categorical feature; state, international plan and the voice mail plan.

Distribution of categorical features

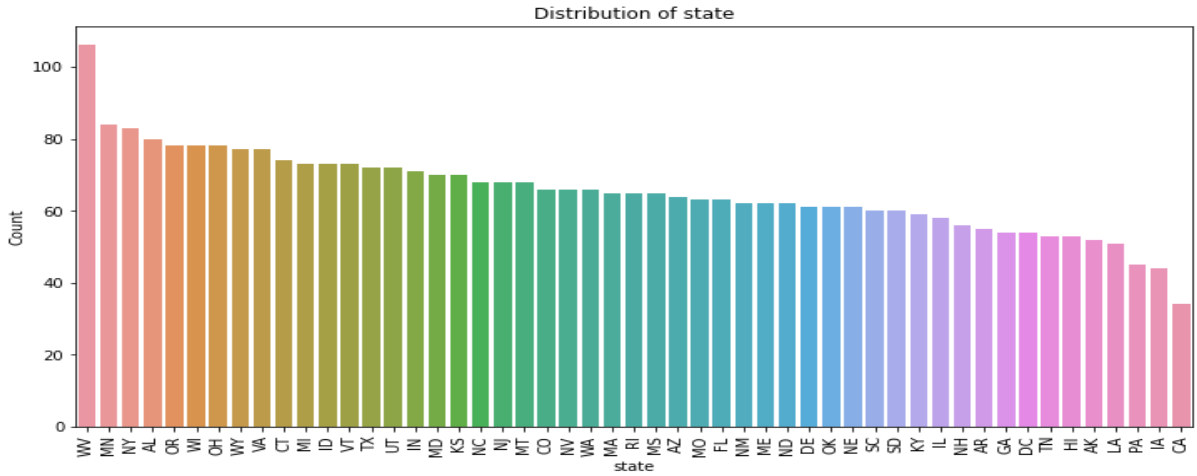

Fig.1: Bar plot (Distribution of customers across states)

We see that the majority of the customer's base is from West Virginia, Minnesota, NewYork, Alabama and Wisconsin. States such as Indiana, penyslvania and California had the least number of customers.

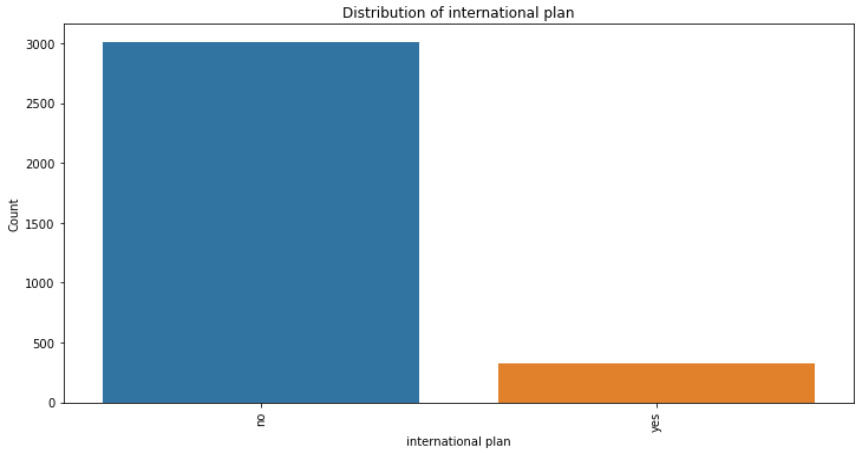

Fig.2: Bar plot (Distribution of customers for the international plan subscription)

Of the SyriaTel customers, only 323 have an international plan.

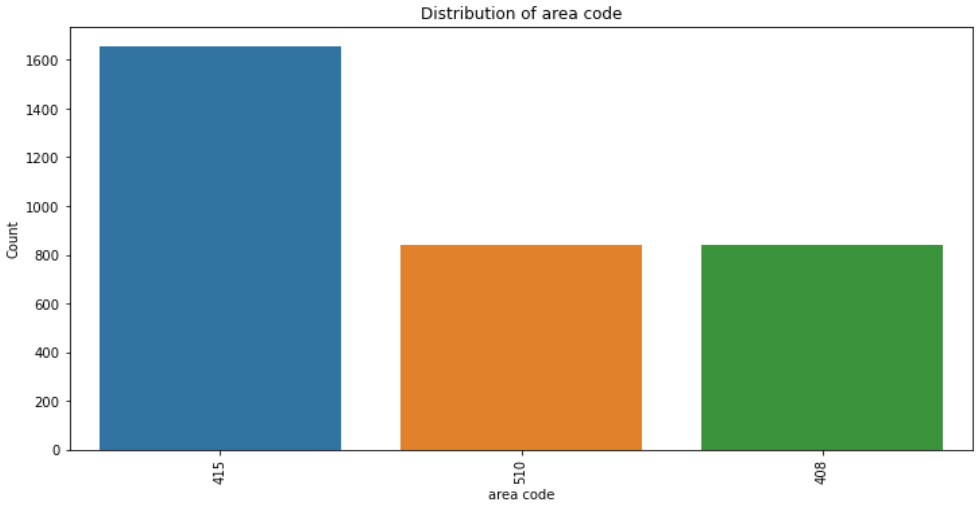

Fig.3: Bar plot (Distribution of customers across area codes)

The distribution of customers across the area codes was almost even for area codes 510 and 408, while it was highest for the area 415. This indicates that area 415 has the most customers, this can be attributed to the possibility that this the area where the company is domiciled. The even distribution across the other two area codes could be influenced by other unknown factors.

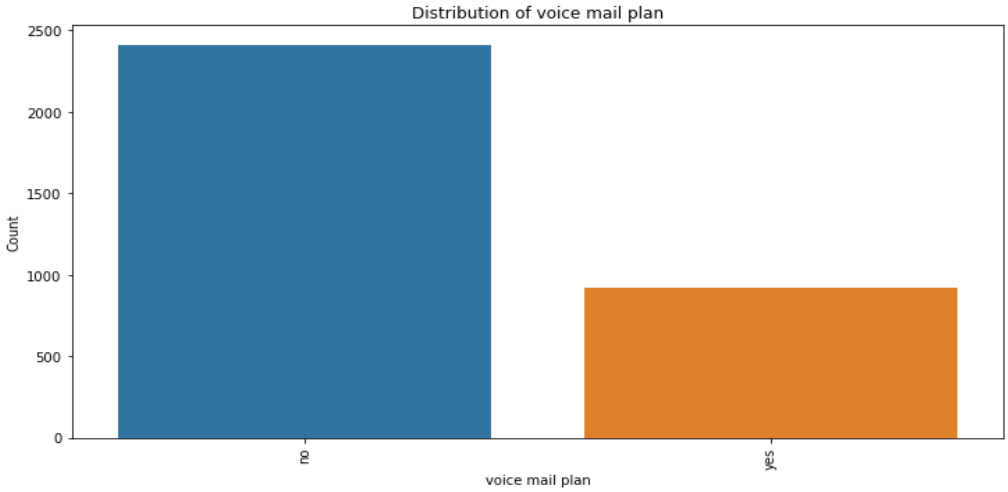

Fig.3: Bar plot (Distribution of customers for the Voicemail plan)

Out of the 3333 customers, only 922 are subscribed to the voice mail plan

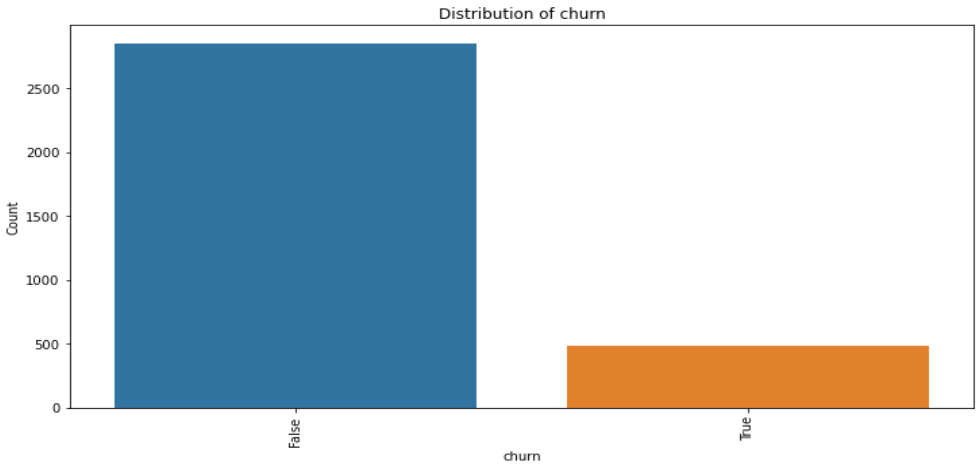

Fig.3: Bar plot (Distribution customer churn)

This is our target feature. Out of 3333, around 14.5% have churned. This is a binary class and the distribution show imbalance that must be checked before we model.

Distribution of numerical feature

For univariate analysis for the numerical features, we defined a function that returned visualization for the distributions of these features. The distribution of the numerical features in the dataset appears to be mostly normal, as shown in the accompanying plots. A normal distribution is typically characterized by a bell-shaped curve, where the majority of data points cluster around the mean, and fewer data points are found as you move farther from the center. This pattern suggests that most of the numerical features, such as customer age or tenure, are evenly distributed with no extreme deviations, making them suitable for modeling without requiring significant transformations.

In terms of categorical features, the distribution of area codes reveals that the majority of customers belong to area code 415, with a smaller, but relatively equal, number of customers from area codes 510 and 408. This suggests that most of the customer base is concentrated in the 415 area code, while the other two area codes contribute a similar number of customers. The visual representation of these area codes likely shows a distinct dominance of the 415 area. International calls, the distribution is slightly skewed, though it still retains a roughly normal shape. The skewness indicates that while most customers make only a few international calls, there is a smaller group of customers who make significantly higher numbers of international calls. This could represent a segment of heavy international callers, who may have different needs or churn behaviors compared to those who make fewer calls. The customer service calls shows several peaks, showing a varying pattern of the population that could be possibly triggered by other factors. These multiple peaks suggest that certain groups of customers contact customer service more frequently, which could be due to specific issues such as technical problems, billing inquiries, or service dissatisfaction.

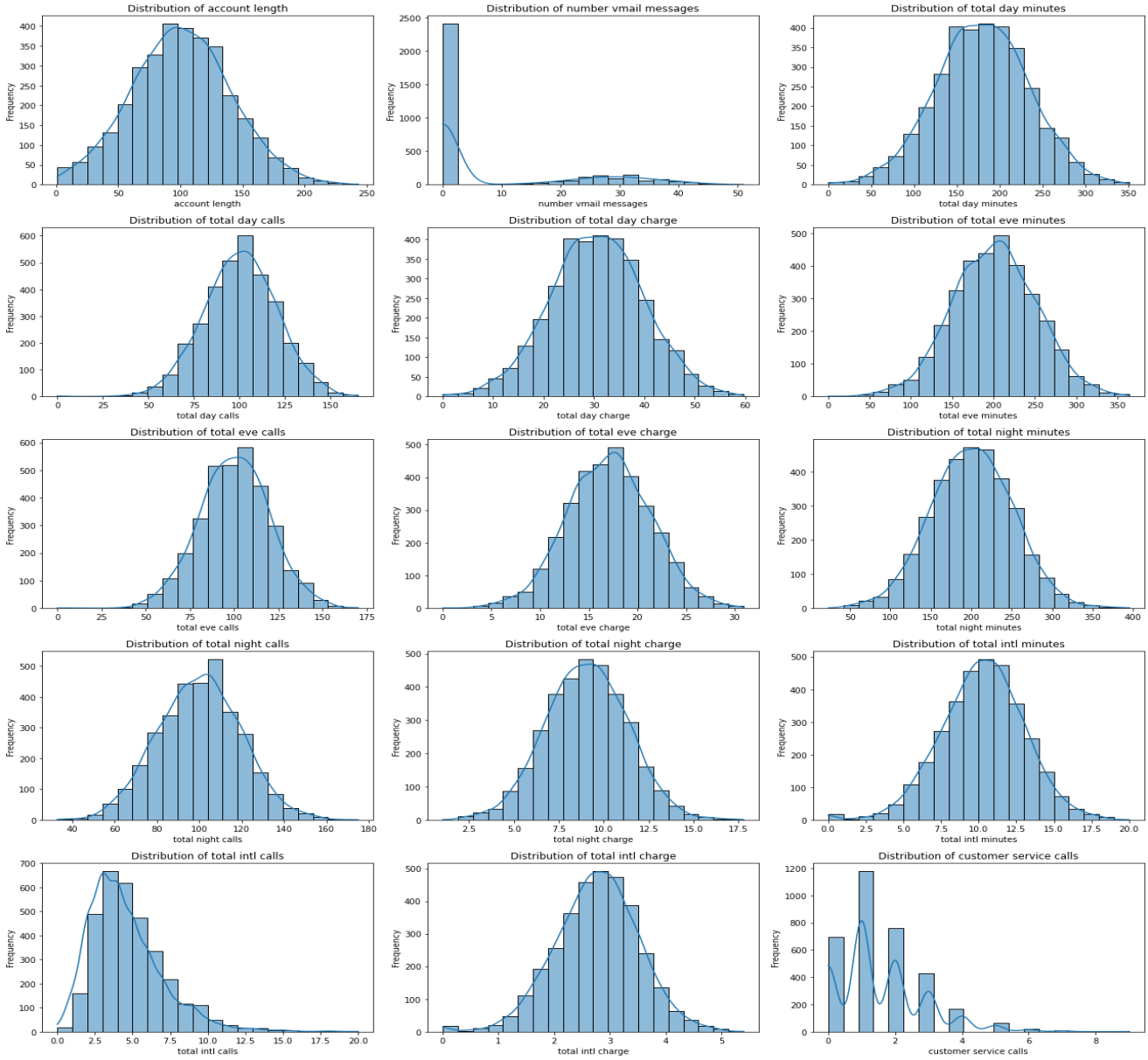

Fig.4: Distribution plots for all the numerical features.

ii. **Bivariate Analysis**

Bivariate analysis entails the use of statistical tools to evaluate relationships between two variables. According to Fancera (2023), bivariate analysis forms the basis for multivariate analysis. In bivariate analysis, we analyze one to one relationships between variables. This will allow us to identify significant interactions or correlations, patterns, dependencies and trends between features in our data set. The goal is to identify changes in variable that are associated with the others. We evaluate the bivariate by analyzing both categorical and numerical, individually against churn.

Categorical bivariate analysis

We define a function that groups by the categorical feature we want to plot for, then retrieve the counts of churn in that group. We then plot for the churn counts of each categorical feature.

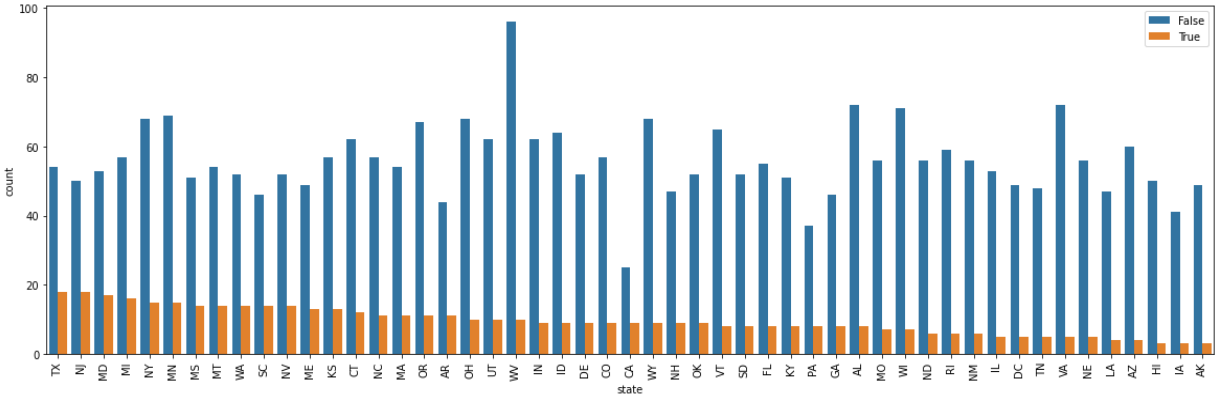

Fig.5: Bar-Plot (Distribution churners and non-churners across states).

The majority of customers who churned were from Texas, New Jersey, and Minnesota as shown by the plot above. Churn rate were low at state AK, IA, HI and AZ

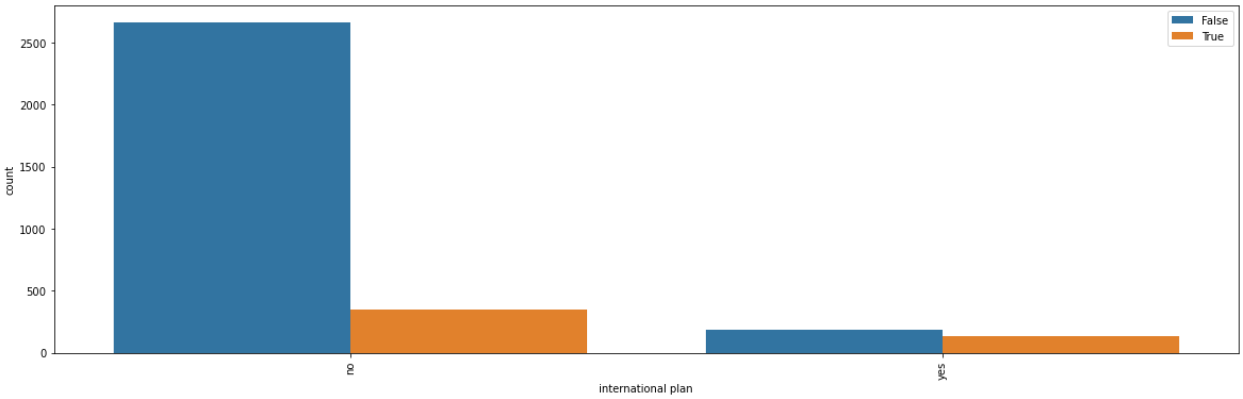

Fig.5: Bar-Plot (Distribution churners and non-churners across international plan).

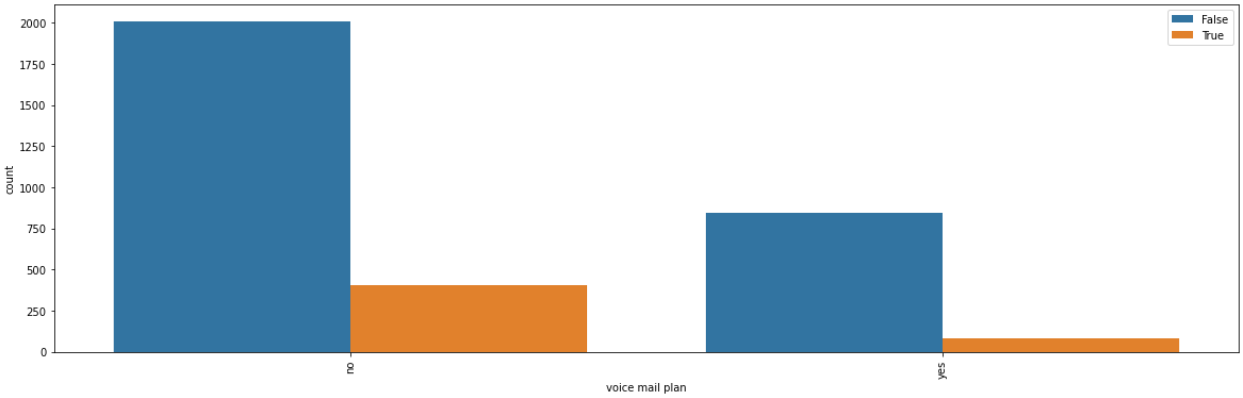

Fig.5: Bar-Plot (Distribution churners and non-churners across voice mail plan).

The plot demonstrates that the majority of churned customers lacked a voicemail or international plan. This implies that users who do not have these plans might use the service less frequently. Lack of these characteristics may increase turnover and decrease satisfaction. It suggests that by enhancing the service's value, providing these options could lower turnover. If other features have an effect on client retention as well, that might be investigated further.

Numerical bivariate analysis

Next, we defined a function that plots the kernel densities of the numerical features. We first merged the numerical data frame with the churn column to allow for bivariate comparison against the churn feature. We then call the function that plots the kernel densities for all numerical features against the churn and interpret.

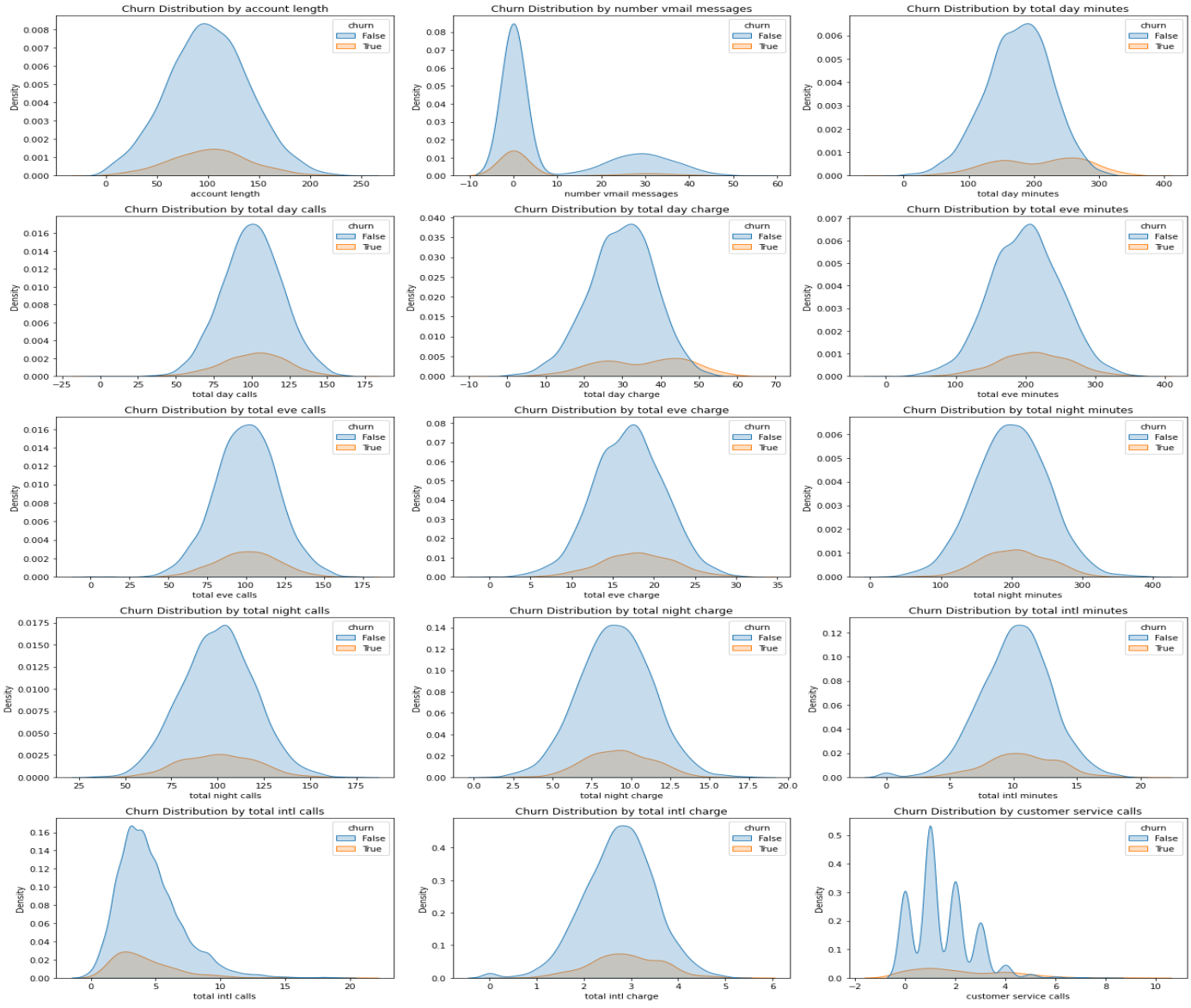

Fig.5: kde plot (Distribution churners and non-churners across all numerical features).

The above plots show that customers who churn typically have much higher charges across a number of categories, including total initial charges, night charges, evening charges, and total day charges. This implies that clients who are charged more are more likely to leave since they may find these fees unaffordable. Remarkably, the proportions of customers who churn vs. those who do not churn across these attributes are fairly balanced, with the former not significantly outnumbering the latter. In order to address the discontent of higher-charge clients before they choose to terminate their accounts, it is crucial to employ proactive customer retention techniques.

iii. Dealing with outliers

Outliers in predictive models can cause overfitting, reduced accuracy, and instability during training. The idea of identifying the outliers is to get those values that does not conform to expected outputs (Sullivan et.al. 2021)).Excluding outliers, which are data points beyond 3 standard deviations from the mean, improves accuracy and stable training. Eliminating outliers aligns with algorithms' assumptions and enhances model performance on unseen data, ensuring a normal distribution. For our case we define the outliers as values lying beyond 1.5 times the interquartile range (IQR). We will excluded values beyond the lower and upper bounds defined by use of the inter-quartile range by filtering through the data frame. The number of rows dropped from 3333 to 2797 rows.

iv. Feature Correlations

Feature correlation shows how changes in one feature are linked to changes in another by measuring the relationship between two or more variables. When two qualities rise or decrease together, there is a positive correlation, and when one feature improves while the other declines, there is a negative correlation. These correlation coefficients are used to quantify these relationships. A correlation coefficient close to 0 suggests no linear relationship between the features. From the correlation values, correlation matrix, we visualize these values using a heat map. They are used to display correlation matrices and represent data using different colors to show the value or intensity of each data point in a matrix. Each cell color exemplifies the strength of the correlation between variables in a heat map (Singh & Nagahara, 2024).
From the heat map, several features show perfect positive correlation as shown in Fig 6.

The perfectly correlated features were;

- Total day charge and total day minutes
- Total initial charge and total initial minutes
- Total night charge and total night minutes
- Total evening charge and total evening minutes**.**

This is approves the rational that charges are directly influenced by total minutes spent. The influence of the perfect correlation impacts the model performance when it comes to its ability to generalize to the unseen data, a phenomenon referred to as overfitting or the dummy variable trap. To deal with this effect of overfitting, we will have to model on one side of the features that lead

to perfect correlations, ie, only use one set of the feature, either the charges or the minutes features. We dropped total initial minutes, total day minutes, total night minutes, total evening minutes.

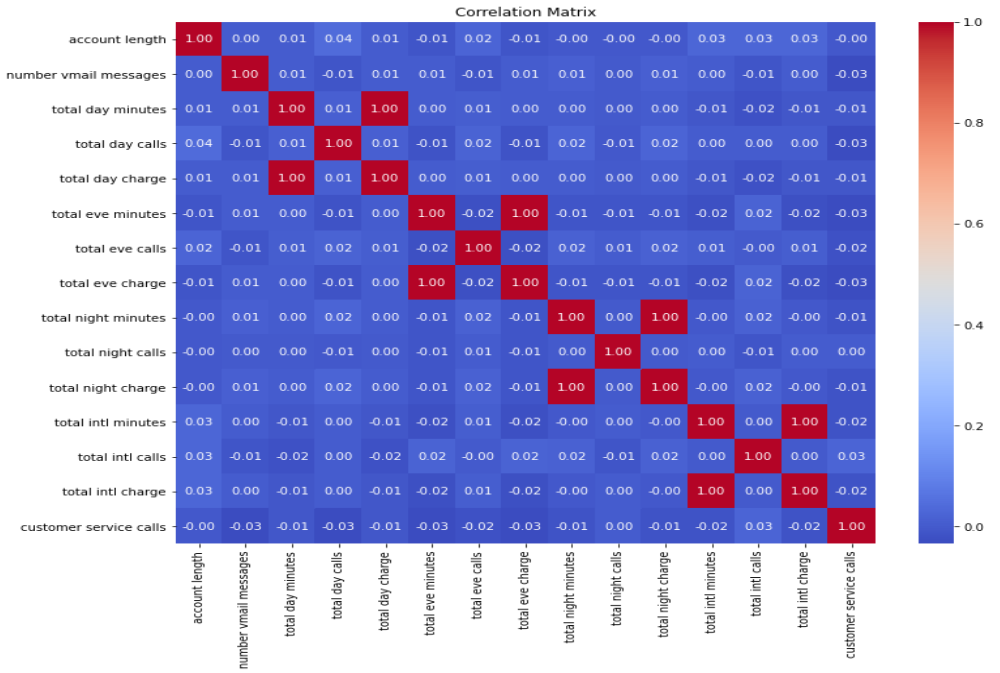

Fig.6: Heat map (correlations between numerical features).

Feature Engineering & Preprocessing

Feature engineering is the process of turning unstructured data into useful features that more accurately depict the underlying issue. Since the nature and quality of the features have a direct impact on the model's capacity to learn and generalize from the data, it is an essential step in the development of machine learning models (Galli, 2024). By transforming unstructured data into a more informative and structured format, feature engineering seeks to increase model performance and accuracy by facilitating the model's ability to identify pertinent patterns. To optimally ensure no issues like data leakage, we first split the data into training and testing sets. If we perform feature engineering on the entire data set, we will allow the testing dataset to influence the training feature's parameters. We do it separately to ensure that the training set does not have the knowledge of test parameters/characteristics. This retains the autonomy of the testing data as truly unseen data. From this point we then perform the suitable feature engineering:

Handle categorical feature by label encoding and one hot encoding and scaling continuous features. We first split the data for modelling by using the train test split, a skit learn function. The train test split function divides the data into two, training data and testing data. We pass in the thresholds for subdividing the data. We used 25% of the data for testing and the 75% for training the model splitting prior to data preprocessing and data engineering reduces the chances of data leakages. We label encoded the international and the voice mail plan columns since they have binary data (yes/no). Label encoding the state would not suffice since the state abbreviations are

themselves unique identifiers. We proceeded to one hot encoding the state and area codes. We preprocessed these data simultaneously for test and train data. This process increased the number of columns from 21 to 65.

The numerical features required standardization to ensure the model performs with the limits of the standard deviation. By standardizing the data, we transform it to have a mean of 0 and standard deviation of 1. The standardization process entails subtracting the mean of the data and dividing by the standard deviation. Standardization is majorly used when the data follows a normal or Gaussian distribution and the data in the features have different scales. The values in the data set features need not be constrained in a specified range, thus standardization is the best transformation to consider.

Modelling

Logistic Regression Model

In this phase we developed a predictive model for customer churn using the features available in our cleaned and preprocessed dataset. The model was assessed based on the success metrics set to correctly identify the customers who are likely to churn. To achieve this, we implemented logistic regression, a widely used algorithm for binary classification tasks, which is suitable for predicting churn. Logistic regression allowed us to assess the probability of customer churn based on the input features and provide interpretable results, which was essential for understanding the key drivers behind customer attrition.

We build a baseline model by invoking the logistic regression from skit learn. We then fit, to train, the model to the X train and y train from the train test split function. By training the model on this data, the model gets to understand the data, thus when presented with the new unseen data, it is capable to predict with certainty. Mostly, the models over trains or over fits to the training data, meaning that it learns every detail in the training data. This is often not desirable because after overfitting it fails to generalize well to new unseen data. Failing to generalize well implies that the predicted values varies a lot from the true value. In over fitted model, the accuracy on testing to the new data is often lower than the accuracy on training data.

When the model is prospected to over fit, we counter overfitting by checking at variety of issues like, class imbalance, and hyper parameters. Class imbalance is tackled through ensuring a balance between the majority and minority data. This is done by either oversampling the minority class or by under sampling the majority class. SMOTE, a skit learn algorithms handles the imbalance by oversampling the minority class to balance the majority class. The model is then retrained on this balanced data and accuracy and other success metrics, area under the curve, precision, recall and f1-score, are crosschecked between the baseline model and the model trained on the new data. If the new model performs better than the baseline model, it is taken as the new baseline model. We are interested in a well-tuned model. Modelling improved by the use different hyper parameters and weightings, a process called regularization or hyper parameters tuning. Finally, the best performing model is picked which is evaluated against the success metrics.

The Baseline Model

In [37]:

```
# import the necessary libraries
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, accuracy_score

# initialize a logistic regression model
baseline_model = LogisticRegression()
# fitting the model
baseline_model.fit(X_train, y_train)
#predictions on the training set
y_train_pred = baseline_model.predict(X_train)

#baseline model evaluation on the training set
train_accuracy = accuracy_score(y_train, y_train_pred)

print(f"Training Accuracy: {train_accuracy}")
print("Training Classification Report:")
print(classification_report(y_train, y_train_pred))
```

```
Training Accuracy: 0.9251311397234144
Training Classification Report:
              precision    recall  f1-score   support

           0       0.93      0.99      0.96      1883
           1       0.77      0.38      0.51       214

    accuracy                           0.93      2097
   macro avg       0.85      0.68      0.73      2097
weighted avg       0.92      0.93      0.91      2097
```

Fig.7: Baseline model training & Accuracy

The training accuracy imply that the model correctly predicts 93% of the samples in the training set. This is considerably high and imply that the model is performing well on training data. The other metrics are also high; recall, precision, and f1 -score. The model shows bias in categorizing or predicting the class in regard to the metrics; precision, recall, and f1-score. We also evaluated the performance of the baseline model on the test data.

```
-------------------------Test Accuracy----------------------------------
0.9014285714285715
-------------------------Testing Classification Report-----------------------------------
              precision    recall  f1-score   support

           0       0.91      0.98      0.95       610
           1       0.76      0.34      0.47        90

    accuracy                           0.90       700
   macro avg       0.83      0.66      0.71       700
weighted avg       0.89      0.90      0.88       700

-------------------------Confusion_Matrix-----------------------------------
[[600  10]
 [ 59  31]]
-------------------------Area_underCurve-----------------------------------
0.8171402550091075
```

Fig.8: Baseline model performance on the test data.

The test accuracy is approximately 90% which is lower than the training accuracy. This imply that the model does not generalize well to the unseen data. The model precisely predicts class 0, 91% of the time and class 1, 76% precision. Recall of 98% on the class 0 and 34% on class 1. The f1 score for class 0 is also considerably high with 95% for class 0 and 47% for class 1. This is a poor performance. Generally, the model performs well on the testing data on class 0, the significant drop in the metrics, precision and recall for class 1 suggests that the model fails to generalize the minority class when presented with the testing data. The baseline model is thus prospected to be overfitting to the majority class in the training data. We can also reduce SMOTE oversampling ie, resampling/ oversampling class 1 and under sample class 0.

Dealing with class Imbalance

We deal with class imbalance in in the training data set and before model training to ensure that the model learns meaningful patterns from both majority and the minority classes. We can oversample the minority class or under sample the majority class, but only on the training set. Use the SMOTE (Synthetic Minority Oversampling Technique) to only generate synthetic samples of the minority class, ensuring balance set.

```
Training Accuracy: 0.938396176314392
Training Classification Report:
              precision    recall  f1-score   support

           0       0.93      0.95      0.94      1883
           1       0.95      0.92      0.94      1883

    accuracy                           0.94      3766
   macro avg       0.94      0.94      0.94      3766
weighted avg       0.94      0.94      0.94      3766

-------------------------Test Accuracy--------------------------
0.88
-------------------------Testing Classification Report----------
              precision    recall  f1-score   support

           0       0.92      0.94      0.93       610
           1       0.54      0.48      0.51        90

    accuracy                           0.88       700
   macro avg       0.73      0.71      0.72       700
weighted avg       0.87      0.88      0.88       700

-------------------------Confusion_Matrix-----------------------
[[573  37]
 [ 47  43]]
```

Fig.8: Baseline model performance on balanced data.

Without class imbalance, we see that the model can equally predict the class zero and 1. Training accuracy on the balanced data set 93 percent, almost equal to that of the imbalanced data set, but here we see great improvement on precision, recall and f1-score metrics. We proceed to use this new baseline model and predict using the X test data. After dealing with the class imbalance and predicting with the test data, we see an improvement on the recall and F1 score at the expense of precision of the minority class and the accuracy of the overall model. The accuracy on the test set is 88% which is significantly lower than the training accuracy, implying the model does not generalize well on unseen data, i.e overfitting. The precision drops from 76% to 54% but the recall and f1 score increases. However, the AUC also drops from 81% to 79%. So we can proceed with the balanced baseline model and try to improve on the same. We can proceed to apply regularization, the l2 regularization for logistic regression.

Model Tuning

Regularization

This technique prevents overfitting by discouraging more complex modes. we prospect that our model is overfitting due to poor generalization to new data. The training accuracy is higher that the test accuracy. To control regularization, we alter the size of the parameter C, a parameter that controls the strength of regularization. This value is inversely proportional to the regularization strength. This implies that;

- larger values of C, results to less regularization, since the penalty term have smaller effect on the cost function
- Smaller values of C leads to more regularization.

The default value of C in logistic regression is typically 1.0. This value represents a balanced or moderate regularization. For our case, at the default regularization, C= 1.0, we still prospect that the model is overfitting on the training data. Thus we need to increase regularization strength by lowering the value of C. We try a second model with altered, lower values of C, and pick the most stable model. Thus, we evaluated the model on various values of C and solvers until we found the best model with the below performance metrics.

```
Training Accuracy (C=0.06, Solver=liblinear): 0.8449283058948487
Training Classification Report:
              precision    recall  f1-score   support

           0       0.84      0.85      0.85      1883
           1       0.85      0.84      0.84      1883

    accuracy                           0.84      3766
   macro avg       0.85      0.84      0.84      3766
weighted avg       0.85      0.84      0.84      3766

-------------------------Test Accuracy----------------------------------
0.8242857142857143
-------------------------Testing Classification Report-----------------------
              precision    recall  f1-score   support

           0       0.95      0.85      0.89       610
           1       0.39      0.67      0.49        90

    accuracy                           0.82       700
   macro avg       0.67      0.76      0.69       700
weighted avg       0.87      0.82      0.84       700

-------------------------Confusion_Matrix-----------------------------------
[[517  93]
 [ 30  60]]
------------------------Area_underCurve---------------------------------
0.7950819672131149
```

Fig.9: Baseline model performance on tuned hyper parameters.

We found that for higher weighting of the minority class 1, the customers who churn, we get overall lower accuracy of the model on the testing set but the area under the curve remain constant. We thus try to strike a balance in the weighting of the minority class to a point where we have considerably higher accuracy and better metrics for the minority class.
By manually adjusting the class weights, the solver and C to the most optimal values;

- C = 0.06
- penalty = 'l2'
- solver = saga
- class weight = {0: 1, 1: 2}

We come to a better model, a better off Recall at the expense of precision and f1 score.

The Final Logistic Regression Model Performance

```
-------------------------Test Accuracy-----------------------------------
0.7257142857142858
-------------------------Testing Classification Report------------------------------
              precision    recall  f1-score   support

           0       0.95      0.72      0.82       610
           1       0.28      0.74      0.41        90

    accuracy                           0.73       700
   macro avg       0.62      0.73      0.62       700
weighted avg       0.86      0.73      0.77       700

-------------------------Confusion_Matrix-------------------------------------
[[441 169]
 [ 23  67]]
```

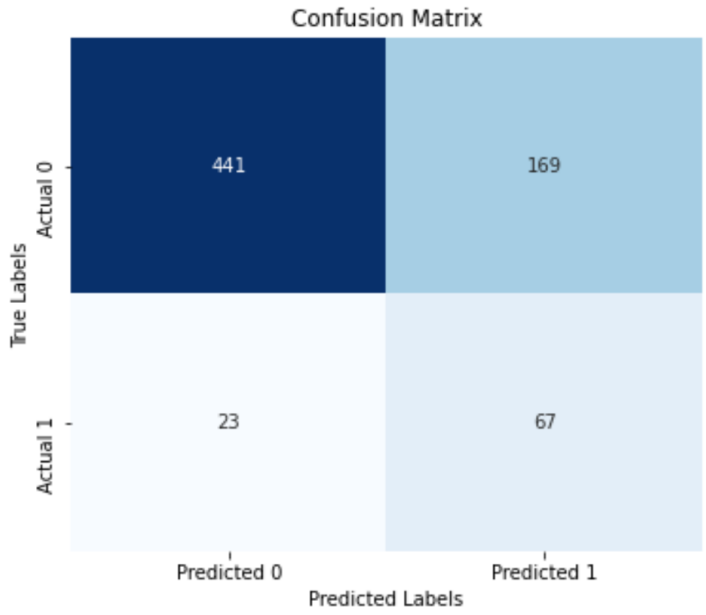

Fig.10: Final model performance (Accuracy, classification Report & confusion Matrix)

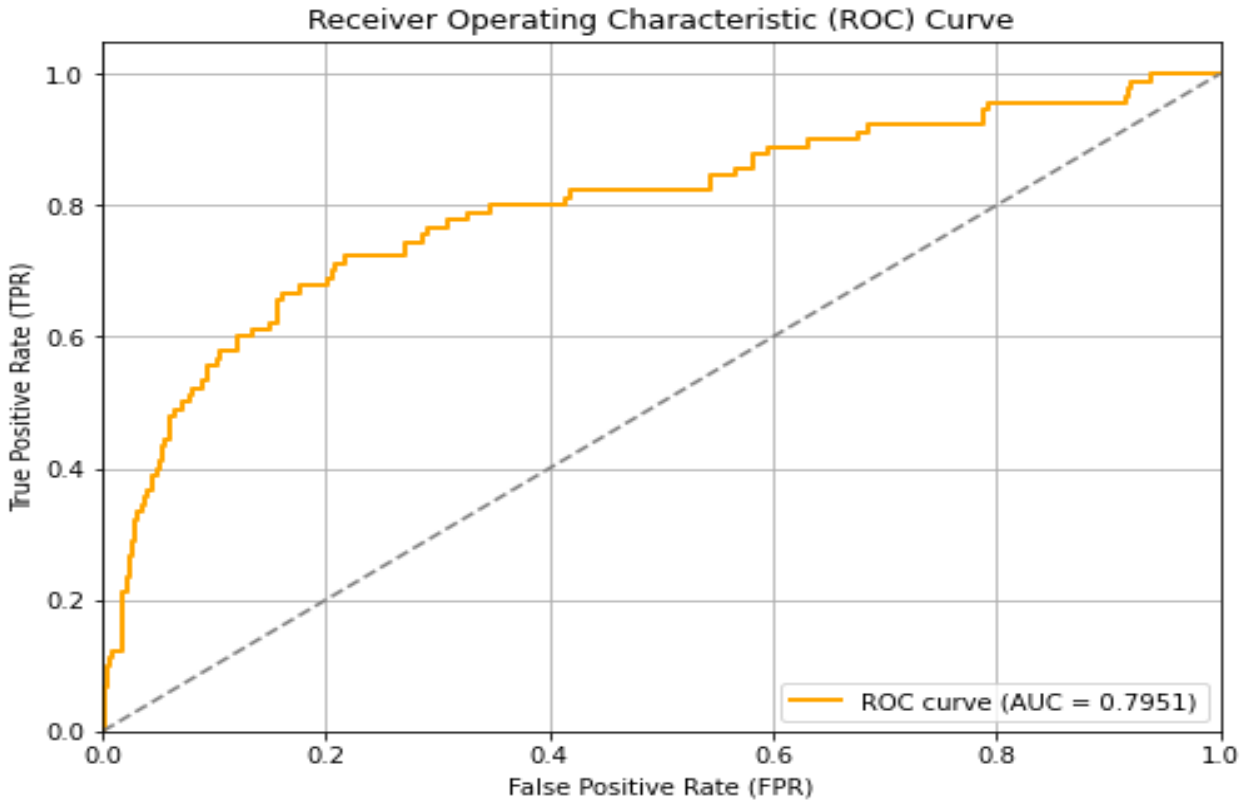

Fig.11: Final model performance (ROC-AUC)

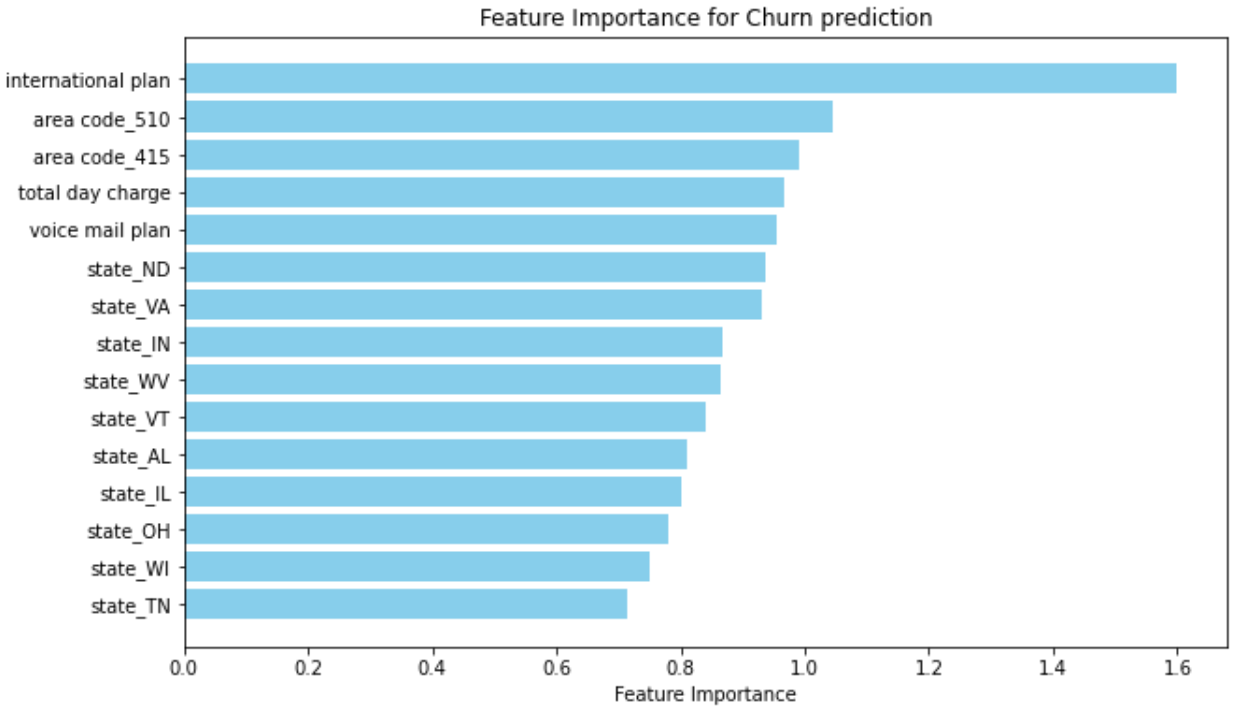

Fig.12: Final model feature importance

Final Logistic Model evaluation

The test accuracy of 0.7257(73%) represents the overall percentage of correct predictions of the customers who are likely to churn. However, this accuracy score does not cover all the instances for when specific class are to be predicted. The classification report gives more details on the model performance, for each class and focusing on the key performance metrics.

- The majority Class (0)
    - Precision of 0.95 - tells that of all the instances the model predicted class 0, it was actually correct 95% of the time.
    - Recall of 0.72 - indicates that the model correctly identified class 0, 72% of the time. It is a moderate value and indicates that the model missed this class 28% of the time.
    - F1 score of 0.82 implies that the model is both accurate at predicting the class 0 and good at finding most of these class as true.
- The minority class (1)
    - Precision of 0.28 imply that the model predicts this class only 28% correctly. The model is definitely not good at predicting this class.
    - Recall of 0.74 tells us that the model correctly identifies class 1, 74% of the time. This is a relatively good score, the model can detect most of the true class 1 customers.
    - F1 score of 0.41 is a relatively lower score indicating the poor balance between the recall and precision of class 1.

The AUC of 0.7951 is relatively a good metric, telling us that the model is better than random guessing. It has a relatively higher capacity to differentiate between the classes. From the feature importance visual, the most important features at predicting the customers who are likely to churn are; International plan, area code 450 and 415, total day charge, voice mail plan and state ND and VA.

The Decision Tree Classifier

After getting low performance metrics of the logistic regression model, we tried the decision tree classifier. The decision tree algorithm splits the data into smaller subsets that are more simplified to contain at least one of the category. In this section we modelled a decision tree and obtained the performance metrics that were compared to the logistic model above. The process of modelling a decision tree classifier is similar to that of logistic regression, class balancing, hyper parameters tuning and final model evaluation.

The training accuracy of 100% showed potential overfitting. The model perfectly fits on the training data which not desired since the model could not generalize well to the training data. The test accuracy dropped to 91% suggesting and proving that the model over fitted to the training data. We thus addressed overfitting by tuning the hyper parameters; maximum depth, minimum sample splits, and sample leafs. Hyper parameters tuning is an important step that permits altering of the hyper parameters and model an optimum performing decision tree classifier. The idea is to find the best combination of these parameters that reduces overfitting and improves generalization to the unseen data.

```
Decision_classifier = DecisionTreeClassifier(random_state = 42, criterion='entropy')
#Fitting/training the model
Decision_classifier.fit(X_train,y_train)
#Lets make predictions on the test data
y_predicted = Decision_classifier.predict(X_train)
# check the accuracy of the model on the training set
accuracy = accuracy_score(y_train,y_predicted)
accuracy
print(f"Training Accuracy: {accuracy}")
print("Training Classification Report:")
print(classification_report(y_train, y_predicted))
```

```
Training Accuracy: 1.0
Training Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      1883
           1       1.00      1.00      1.00       214

    accuracy                           1.00      2097
   macro avg       1.00      1.00      1.00      2097
weighted avg       1.00      1.00      1.00      2097
```

Fig.13: The Baseline Decision Tree Classifier Model

Final Decision Tree Model

Using the combination of the hyper parameters that optimize the performance, we model the final Decision Tree Classifier.

```
Decision_classifier2 = DecisionTreeClassifier(random_state = 42,
                                              criterion='entropy',
                                              max_depth = 9,
                                              min_samples_split =30,
                                              min_samples_leaf= 5)
#Fitting/training the model
Decision_classifier2.fit(X_train,y_train)
#Lets make predictions on the test data
y_predicted = Decision_classifier2.predict(X_train)
# check the accuracy of the model on the training set
accuracy = accuracy_score(y_train,y_predicted)
accuracy
print(f"Training Accuracy: {accuracy}")
print("Training Classification Report:")
print(classification_report(y_train, y_predicted))
```

```
Training Accuracy: 0.9656652360515021
Training Classification Report:
              precision    recall  f1-score   support

           0       0.97      0.99      0.98      1883
           1       0.91      0.74      0.81       214

    accuracy                           0.97      2097
   macro avg       0.94      0.86      0.90      2097
weighted avg       0.96      0.97      0.96      2097
```

Fig.14: The Baseline Decision Tree Classifier Model (Tuned hyper parameters)

```
-------------------------Test Accuracy-----------------------------------
0.9442857142857143
-------------------------Testing Classification Report-------------------------------
              precision    recall  f1-score   support

           0       0.95      0.99      0.97       610
           1       0.87      0.67      0.75        90

    accuracy                           0.94       700
   macro avg       0.91      0.83      0.86       700
weighted avg       0.94      0.94      0.94       700

-------------------------Confusion_Matrix--------------------------------
[[601   9]
 [ 30  60]]
-------------------------Area_underCurve---------------------------------
0.8171402550091075
```

Fig.15: performance metrics of final Decision Tree Classifier

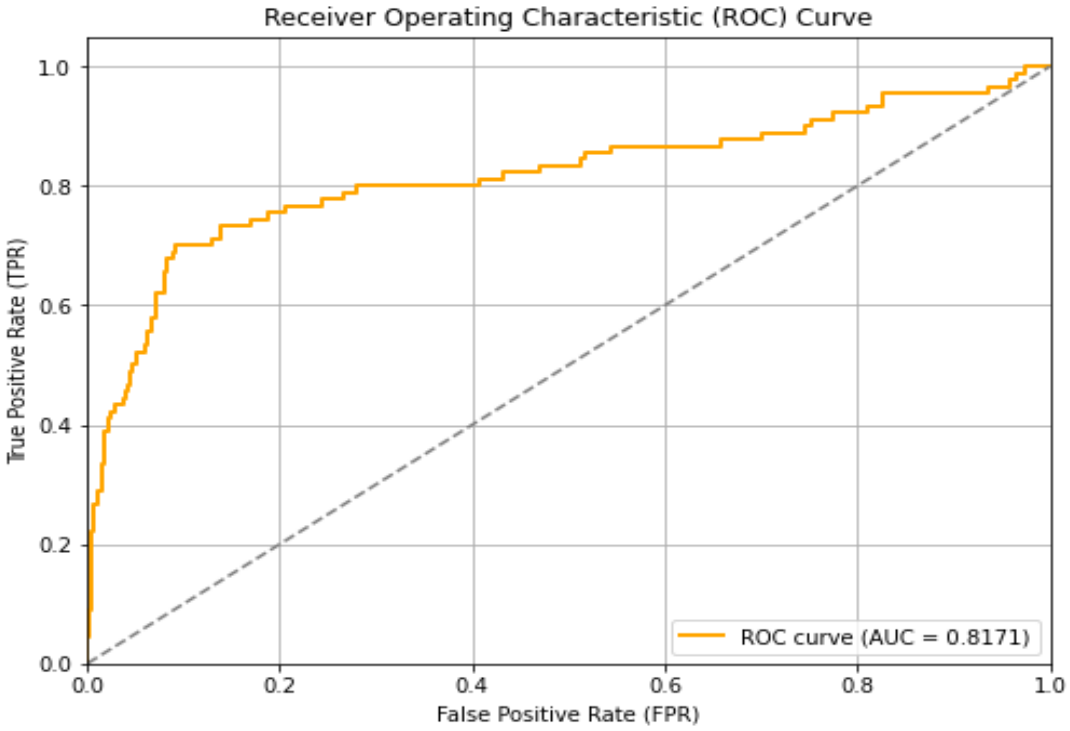

Fig.16: Decision Tree Classifier (ROC-AUC)

Feature Importance of the Decision Tree Classifier

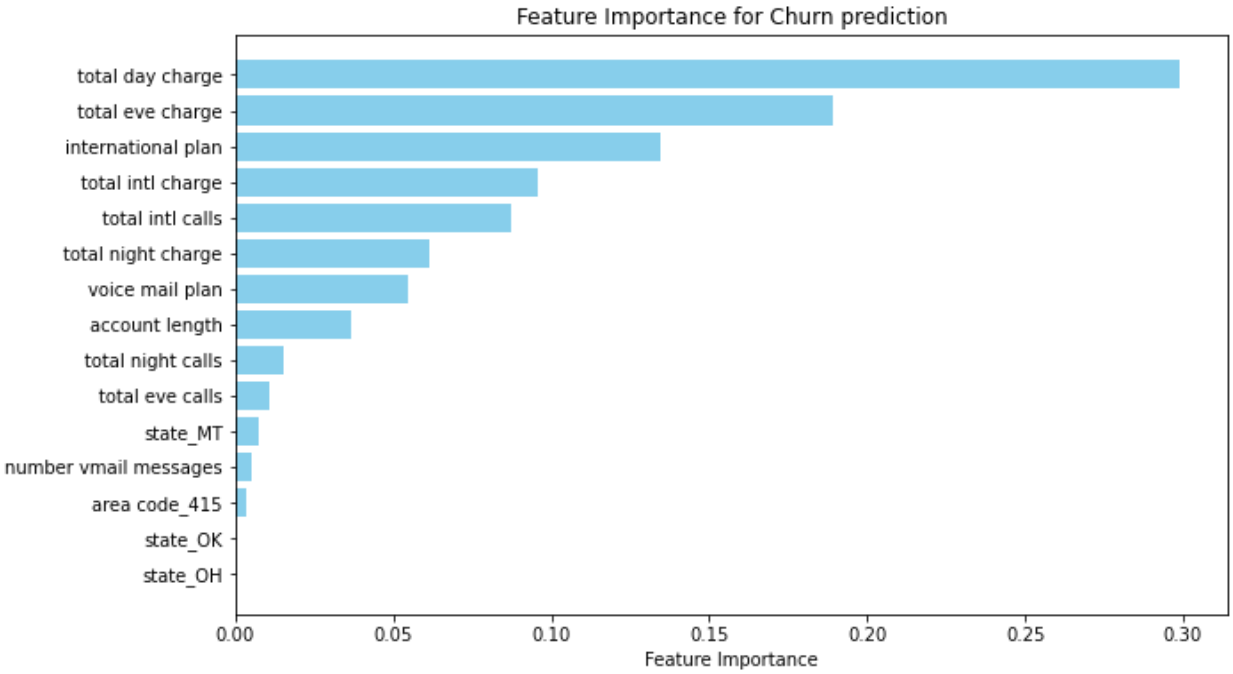

Fig.16: Decision Tree Classifier (feature importance)

Final Decision Tree Model Evaluation

The tuned model shows reduced overfitting since the training accuracy is lower at 96.5% as compared to the baseline model (100%). The model Test accuracy also improved from 91% to 94%. The final model correctly predicted 94% of the new data set(the X_test data). This is a commendable result indicating that the model has been trained well, not overfitting and generalizes well to unseen data.

The classification report:

The majority class (0)

Precision = 95%. Of all predicted instances, 95 % were correctly classified as class 0.

Recall= 99%. Of all predicted instances for class 0, the model was 99% accurate.

F1-score = 0.97. This is the harmonic mean of recall and precision. It indicates that the model performs well in correctly classifying class 0.

The minority class (1)

Precision = 0.87. Of all instances the model predicted class 1, it was correct 87% of the time.

Recall = 0.67. The model correctly classified 67% of class 1. This implies that it missed 33% of this class

F1-score = 0.75. The harmonic mean of recall and precision for class 1. This is moderate performance in predicting class 1.

The confusion matrix:

Class 0:

True negatives = 601
False positives = 9

Class 1:
False negatives = 30
True positives = 60

The AUC of 0.8217 is relatively a good metric, indicating the model's significance over random guessing. It has a relatively higher capacity to differentiate between the classes 1 and 0. In the decision tree classifier, the most important features at predicting the customer churn are; total day charge, total evening charge, international plan, total intl charge and total intl calls.

Conclusions

We achieved the project objectives. We developed two classification models, logistic regression and decision tree classifier. We evaluated their performance metrics; precision, recall and f1-score. By considering the performance metrics of our developed models, the decision tree classifier is preferred for its better performance.

Evaluating Decision Tree Classifier against the success metrics

- Accuracy 94%.
- Precision 87%.
- Recall 67%
- F1-score 75%

The selected model outperformed the success metrics, indicating that the model's performance was superior. Therefore, the Decision Tree Classifier emerged as the most effective choice for this task. The factors that most influenced Decision tree classifier were; total day charge, total evening charge, international plan, total intl charge and total intl calls. From the bivariate analysis, we found that an increase in; total day charge, total evening charge, total intl charge lead to higher rate of churning. From the best logistic model, the most important features at predicting the customers who are likely to churn are; International plan, area code 450 and 415, total day charge, voice mail plan and state ND and VA.

Business Recommendation

Give promotional offers to customers from the area codes 510 and 415. These areas have a high churn rate. Offering promotions such as discounts can act as incentives that discourage customers from churning.

Improve the pricing strategies in favor of the customers, this would include leveraging and adjusting prices for the day, evening and night calls. Also the company can introduce packages that encourage more calls at lower costs for customers who are likely to churn.

Focus on customer retention strategies, for customers having the international plan. These customers have a higher likelihood of churn according to the decision tree classifier and retaining these customers would save the company the cost of acquiring new customers.

Enhance the quality of customer service and decrease the volume of customer service calls by strengthening training programs for customer service representatives. This will ensure quicker and more efficient resolution of customer issues, ultimately boosting customer satisfaction and reducing churn.

Next Steps & Future Improvements

Model validation and comparison: Perform cross validation to check the performance of the decision tree classifier, to validate the accuracy of the model. Comparing the model with other classification models like the XG boost, random forest and SVM using same metrics.

Model improvement: include pruning to reduce the complexity of the decision tree classifier

Model deployment/ Monitoring. If the model is rendered fit for production, it should be deployed and performance tracked overtime.

Actionable Recommendations. Implementing the recommendations and monitor the model performance based on the adjustments.

References

Essler, M. (2023). Univariate Analysis: Variance, Variables, Data, and Measurement. In: Social Science Research in the Arab World and Beyond. SpringerBriefs in Sociology. Springer, Cham. https://doi.org/10.1007/978-3-031-13838-6_2

Fancera, S. F. (2023). Bivariate analysis. Research Design and Methods for the Doctor of Education in Leadership at William Paterson University.

Galli, S. (2024). Python feature engineering cookbook. Packt Publishing Ltd.

ResearchGate, 2020.Predicting customer churn with machine learning algorithms in the telecommunications sector.

Simplilearn (2023) Churn Analysis: Techniques to Retain More Customers. Simplilearn. https://www.simplilearn.com/churn-analysis-article

Singh, N. K., & Nagahara, M. (2024). LightGBM-, SHAP-, and Correlation-Matrix-Heatmap-Based Approaches for Analyzing Household Energy Data: Towards Electricity Self-Sufficient Houses. Energies, 17(17), 4518. https://www.mdpi.com/1996-1073/17/17/4518

Sullivan, J. H., Warkentin, M., & Wallace, L. (2021). So many ways for assessing outliers: What really works and does it matter?. Journal of Business Research, 132, 530-543.

Tracxn.(2024).Syriatel –About the company. Tracxn. https://tracxn.com/d/companies/syriatel/__Catsiw22eTWkAaK6EWH1njbGhSiXoDKWEd7WaF9xvLI#about-the-company