Predicting Customer Churn in the Telecommunications Industry:

A Data-Driven Approach to Enhance Customer Retention at Vortex

Name: Vinay Beesa Gnaneshwar

Student ID: 301335423

Table of Contents

Executive Summary

In today's economic market, there is fierce competition amongst all organizations and in particular the telecommunications industry. Now more than ever is the need to retain customers for this is directly correlated to a company's profitability and longevity in the competitive market. Growth and profitability are directly a by-product of the ability of companies to retain customers.

Vortex, a telecommunications provider who primarily operate in the California market is grappling with high customer churn rates, which is posing as a serious threat to its revenue, market share as well as brand image amongst their fellow players in the telecommunications industry. Reichheld and Sasser (1990) in their Harvard Business Review discuss the importance of reducing defections, they state that by generating just a 5% increase in business or a 5% decrease in churn, companies can generate up to 85% profits. This goes to show the importance of identifying, tackling and mitigating churn for Vortex.

The study looked at several machine learning models, including Logistic Regression, K-Nearest Neighbours, Neural Networks, and Random Forests. Among these, the Random Forest model emerged as the most dependable, providing the best balance of accuracy and predictability. The model's strength is its ability to reduce overfitting and detect complex patterns in data, making it an effective tool for predicting customer churn.

The Random Forest model identified several critical factors that influence churn, including contract type, number of referrals, and internet service type. Customers on month-to-month contracts, for example, are more likely to leave than those with longer-term contracts. Similarly, customers who have multiple service lines or use fiber-optic internet services are less likely to leave.

Based on these findings, the report recommends a number of strategies for reducing churn and improving customer retention. These include targeted retention campaigns for high-risk customers, such as offering incentives to switch to longer-term contracts or upgrade to fiber-optic service. Additionally, improving customer service, especially for those who have expressed dissatisfaction, is critical to customer retention.

Vortex can expect to significantly reduce customer churn by implementing these strategies, guided by the Random Forest model's insights. The model's predictive capabilities also enable continuous improvement as new data is collected, ensuring Vortex's competitiveness in the dynamic telecommunications market.

Overall, this data-driven approach equips Vortex with the tools it needs to retain a loyal customer base, improve customer satisfaction, and ensure long-term profitability.

Executive Objective

The objective of the entire project was to create a predictive model which would help us identify high risk customers who churn due to various factors. By analysing historical churn data, we can better understand what causes churn and implement steps to mitigate churns in the future. Especially in such a high competitive landscape such as the telecommunications industry, customers who churn and especially churn and associate with a competitor, poses a significant challenge in our profitability and growth. For Vortex, understanding this and mitigating churn is paramount in maintaining a stable customer base and in return securing long term success. By leveraging historical customer data, the project aims to equip Vortex with deep insights on addressing customer attrition.

The predictive model developed in this project draws from a wide pool of data points that were collected from Vortex’s existing customer base. Numerical variables such as age, total monthly charges, tenure in months and categorical variables such as dependents, reference, contract type, payment method and so much more were analysed to provide Vortex with actionable insights. These insights are crucial in designing retention strategies that could help mitigate churn and put a cap on the growing churn rate. This in-turn will improve overall customer satisfaction as well.

The insights generated by the predictive model would be crucial in Vortex’s quest to design targeted retention strategies. For example, by identifying customers who exhibit behaviours associated with the likelihood of high churn, Vortex can implement proactive measures such as personalized offers, incentivise early renewals, improve customer service and mush more. These strategies when implemented well will enhance customer satisfaction and reduce the likelihood of customer churn. Additionally, the model can help Vortex allocate its resources more efficiently by focusing customer retention strategies on high-risk customers, rather than applying a one-size-fits-all approach across the entire customer base.

Additionally, by utilizing historical data, the prediction model may be continuously improved. The model can be modified and improved over time as new data is gathered to boost its efficacy and accuracy. By implementing an iterative strategy, Vortex can keep ahead of its competition and maintain a devoted customer base by adapting to changing market conditions and customer behaviour.

Executive Model Description

The dataset was exposed to a number of machine learning models: Logistic Regression, K Nearest Neighbour, Neural Network using MLP-Classifier and Random Forest. Ultimately Random Forest was selected as it provided an excellent balance between predictability and accuracy. The initial stage of the project comprised of initial steps in the project consisted of data pre-processing to ensure a clean dataset. This stage involved handling missing values, normalizing numerical features, and encoding categorical variables in order to produce a clean and consistent dataset that the models could use to make predictions.

Key performance metrics, including accuracy, precision, recall, and overall ROC-AUC scores, were used to assess each model. By examining these metrics, a thorough evaluation of how well each model would forecast customer attrition was possible. Analysing the models was essential to determining the advantages and disadvantages of every machine learning mode.

Executive Recommendations

Taking into consideration Random Forest’s interpretation of the dataset and categorizing features that we need to focus on for improvement, the following are not an exhaustive list but a place to start off our implementations to work at reducing customer churn and enhance customer retention and satisfaction.

1. Targeted Retention Campaigns: The model has identified specific customers to be high-risk, thus addressing this by targeted retention campaigns would be best. Personalized offers such as discounts, enhanced service packages, loyalty points and much more can be used to effectively retain customers.
2. Enhanced Customer Support: Satisfaction scores and churn reasons revealed dissatisfaction with customer service to be a significant factor in churn. Vortex must prioritize in improving its customer service by implementing many strategies for a boost. Successfully implementing this will yield in an enhanced customer service experience and improve satisfaction. Effective training of customer service representatives who can effectively address issues with all customers.
3. Flexible Contract Options: Random Forest model revealed that our customers on a month-to-month contract are more susceptible to churn as compared to those who are on a longer-term contract. In order to mitigate this, Vortex needs to offer attractive incentives for customers who switch to a longer-term contract.
4. Expanding predictive modelling: Exploring other models, such as XGBoost and SMOTE, may yield new insights that improve Vortex's ability to predict and prevent customer churn. These combined strategies aim to improve customer retention and satisfaction, ultimately driving Vortex's long-term growth and profitability.

Introduction - Background

The telecommunications industry is extremely competitive, with companies looking to outperform one another constantly in the quest to maintain and grow their customer bases. However, customers who churn pose a significant challenge to Vortex which leads to loss in revenue and increases costs of new customer acquisition which is typically five to six times higher than retaining existing ones (Farris, Bendle, Pfeifer, & Reibstein, 2010).

For Vortex, customer retention is extremely important as the company faces stiff competition in the California market. Understanding what drives customer churn and implementing effective retention strategies is imperative for sustainable profitability and growth.

Problem Statement

Since customer churn is a critical component in Vortex's profitability, the fix to be in place has to be robust to tackle multiple challenges it poses. Traditional retention strategies have proven to be inadequate in addressing the reasons why customer churn. The challenge is to develop a more robust and sophisticated approach that leverages data to predict churn:

The problem can be summarized as follows:

- High customer churn is leading to significant revenue losses for Vortex.
- There is a lack of insights into specifics of what drives these churns.
- Existing retention strategies are insufficient to handle individual customer needs, leading to missed opportunities.

Objectives & Measurement

The objective of this modelling exercise is to develop a robust predictive model that efficiently and accurately identifies customers from historical data who are at risk of churning. This information can help build a typical persona of a high-risk churn individual for further study. As discussed earlier, in the competitive telecommunications industry, understanding and mitigating customer churn is imperative for a successful business model and securing long-term profitability. According to Reichheld and Sasser (1990), "by generating just a 5% increase in customer retention, companies can see up to an 85% increase in profits" (p. 3). This machine learning model will act as a crucial tool for Vortex by providing insights needed in shaping effective customer retention strategies, which would help in reducing churn rates as well as increasing customer lifetime value (CLV).

Let's now go over what metric we shall be covering and comparing in this project. There are several key performance indicators (KPIs) that are designed to measure different aspects of a model's performance. Below is a summary of these KPIs and what they entail:

- Accuracy: Accuracy represents the proportion of correctly identified churn vs non-churn classes. The main purpose of an accuracy score is to provide an overall measure of how well the model performs and predicts. Accuracy is an important metric as it tests both measures of prediction in terms of whether the model can accurately predict non-churn to be non-churn and churn to be churn.
- Precision: The percentage of real churns, or true positives, among all of a model's positive predictions is known as precision. Stated differently, precision quantifies the model's capacity to accurately detect churn cases while reducing false positives. This metric is especially crucial when it comes to preventing needless retention expenses and efforts by incorrectly classifying non-churn cases as churns. A high precision guarantees that a model makes confident churn predictions.
- Recall: A model's recall gauges how well it can isolate real churn cases from all of the customers who actually left. Recall is a crucial metric in assessing a model's capacity of

predicting true positives. A high recall usually indicates whether a model is effectively flagging high-risk customers.

- F1-score: The F1-score is a metric that provides a balance between precision and recall by taking the harmonic mean of both metrics into account. A F1-score helps in evaluating a model's overall performance by ensuring that neither precision nor recall is disproportionately weighed.
- ROC-AUC Score: The curve, also known as the Receiver Operating Characteristic, shows the true positive rate at different threshold intervals plotted against the false positive rate. This curve gives us a thorough understanding of how well a model performs across various thresholds. The model's discriminative ability is typically indicated by a higher Area Under the Curve (AUC), which summarizes the performance in a single value. One important measure of the model's efficacy in a binary classification setting is the ROC-AUC score, which is a reliable metric for evaluating the model's overall ability to discern between churn and non-churn cases.

Assumptions and Limitations

The analysis carried out in this project is grounded by several assumptions that shape the overall approach and its outcomes. These assumptions are critical in providing an understanding of the scope and applicability of the predictive model:

- Data Representation: It is assumed that the dataset used in modelling reflects Vortex's entire customer base. In other words, the sample data of 7,000 customers capture the broad spectrum of customers that Vortex has and in-turn captures the broad spectrum of customer characteristics and behaviours. This assumption is crucial as the model's predictions are based on patterns observed in the dataset. If the data is not representative, the model may fail to generalize effectively to the wider customer population, leading to inaccurate predictions.
- Consistency of Churn Factors: The assumption of factors that influence customer churn is to remain constant over time. Which means that all of the variables identified as predictors of customer churn in the historical dataset is to continue influencing churn in the future in the similar fashion. This assumption allows the model to use historical data to predict future outcomes.

- External Factors: The multitude of external factors like economic conditions, regulatory policies, competition and other key factors remain constant as it did for the historical dataset. This assumption simplifies the models ability to predict, isolating it from future changes in the multitude of external factors that may or may not influence the business of Vortex.

Data Set Introduction

The dataset used to analyse customer churn for Vortex includes 5 different datasets which were then merged into one. Let’s review the main pieces of information that were derived from each dataset without delving too deep.

- CustomerChurn: This dataset focus on the core customer information such as their contract length, usage of various services, billing information and so on which is crucial in identifying churn behaviour.

Column Name	Description
Customer_ID	Unique identifier.
Senior Citizen	Indicates if customer is a senior citizen.
Dependents	Shows if customer have dependents.
Tenure	The no. of months a person is a customer of Vortex.
Phone Service	Indicates if the customer has a phone service.
Internet Service	Type of internet service the customer subscribes to (e.g., DSL, Fiber).
Monthly Charges	Monthly fee charge incurred by customer.
Total Charges	Total amount charged to date.
Churn	An indicator showing if the customer has churned or not.

- Telco_customer_churn: This dataset adds to the above by providing more information on customers such as their demography information.

Column Name	Description
Country	Country of residence.
State	State in the country.
City	City of Residence.
Zip Code	Postal Code.
Churn Score	A pre-calculated score indicating the likelihood of churn.

- Telco_customer_churn_demographics: This dataset provides more information on customers such as gender, age, their marital status and so on.

Column Name	Description
Gender	Customer Gender
Age	Age of customer.
Marital Status	Marital Status of customer
Number of Dependents	Number of dependents of the customer.

- Telco_customer_churn_location: This dataset provides good information on locations of customers, including coordinates.

Column Name	Description
City	City of residence.
Latitude	Latitude.
Longitude	Longitude.

- Telco_customer_churn_status: This dataset captures churn status of customers, their satisfaction scores and also churn reasons.

Column Name	Description
Customer Status	Indicates whether the customer is currently active or churned.
Satisfaction Score	The customer's satisfaction score.
Churn Reason	The reason for churn if applicable.

Exclusions

An important stage in the dataset preparation process was to exclude specific columns that were regarded redundant, irrelevant or potentially damaging to the predictive model's accuracy and fairness. The choice to eliminate these columns was based on data analysis, subject expertise and best practices in machine learning. This section includes a full review of the columns that were eliminated, as well as the reasoning behind each deletion.

- Initial Exclusions before Data Visualizations:

Certain columns were identified as not contributing meaningfully to the analysis and were therefore excluded early in the process:

Column Name	Reason for Exclusion
Customer_ID	Does not contribute to predictive modeling.
LoyaltyID	Another form of ID, redundant for analysis.
Service ID	Another form of ID, redundant for analysis.
Status ID	Another form of ID, redundant for analysis.
Zip Code	Too granular, with over 800 unique values, not relevant to churn.
Churn Reason	Duplicates information in `Churn Reason_df1`.
Lat Long	Unnecessary, as individual latitude and longitude columns exist.

- Excluding prior to modelling activities:

Post data visualizations were completed, additional columns were identified that could be problematic during the modelling phase. These columns were excluded as it would potentially cause issues and biases in the models.

Column Name	Reason for Exclusion
Country	Single country, no variability.
State	Single state, no variability.
City	Over 850 unique cities, too granular.
Latitude	Too granular, not relevant for modeling.

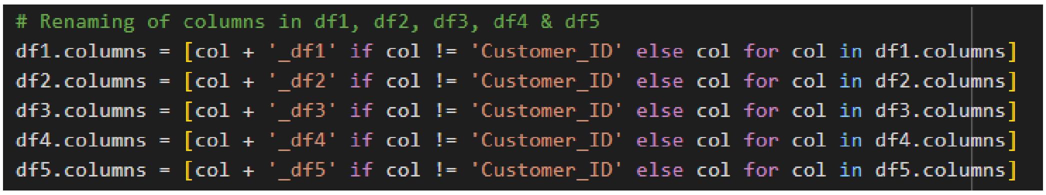
Longitude	Too granular, not relevant for modeling.
Churn_Score	Derived from a previous model, potential for bias.
Churn_Reason	Derived from a previous model, potential for bias.
Churn_Category	Derived from a previous model, potential for bias.
Satisfaction_Score	Makes modeling too easy, directly correlates with churn.
Total_Long_Distance_Charges	Could introduce bias, as charges are strong indicators of churn.
Total_Charges	Could introduce bias, as charges are strong indicators of churn.
Total_Extra_Data_Charges	Could introduce bias, as charges are strong indicators of churn.
Total_Revenue	Could introduce bias, as charges are strong indicators of churn.
Total_Monthly_Charge	Could introduce bias, as charges are strong indicators of churn.
Total_Long_Distance_Charge	Could introduce bias, as charges are strong indicators of churn.

Initial Data Preparation

The initial data preparation was carried out to ensure that the five different datasets were properly merged, structured and had very little if not none inconsistencies, which would facilitate a smooth and robust data analysis via final modelling. The process of data preparation was subjected to merging of datasets, renaming variables for keeping things consistent, removing of duplicate columns, adjusting data in a few columns and finally dropping unnecessary columns that could cause bias in modelling, which have been highlighted in the previous section. Below are in detail descriptions of all steps taken to prepare the data.

1. Renaming Variables for Clarity

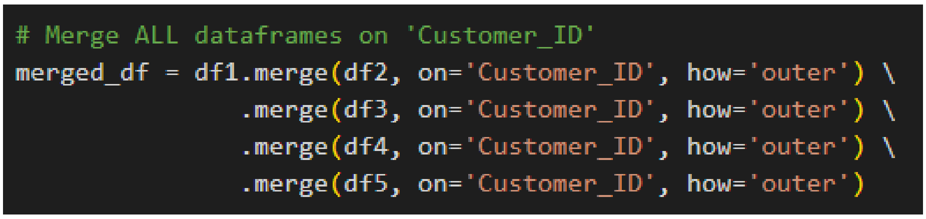
Owing to multiple duplicates found in the five datasets. First all columns were renamed to include a suffix with '_df1', from the second with '_df2', and so on up to '_df5' to ensure that we could track the origins of each variable. This was particularly helpful when we had to compare and drop duplicate columns in the future.



```
# Renaming of columns in df1, df2, df3, df4 & df5
df1.columns = [col + '_df1' if col != 'Customer_ID' else col for col in df1.columns]
df2.columns = [col + '_df2' if col != 'Customer_ID' else col for col in df2.columns]
df3.columns = [col + '_df3' if col != 'Customer_ID' else col for col in df3.columns]
df4.columns = [col + '_df4' if col != 'Customer_ID' else col for col in df4.columns]
df5.columns = [col + '_df5' if col != 'Customer_ID' else col for col in df5.columns]
```

2. Merging of datasets

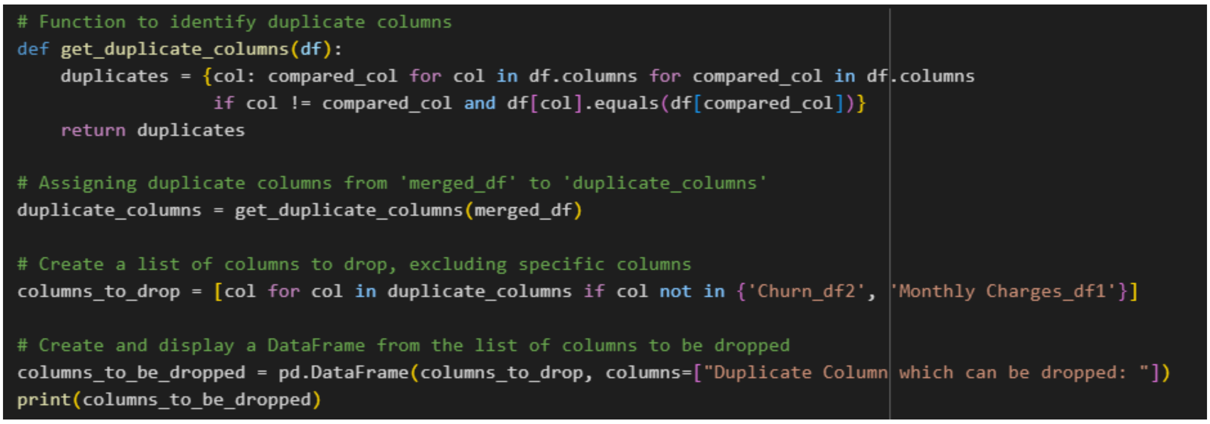
The first step involved merging of the five datasets which housed various types of customer information, with 'Customer_ID' serving as the common field which links all datasets. The 'merge' function in Python's 'pandas' library was used to merge the datasets into a unified data frame.



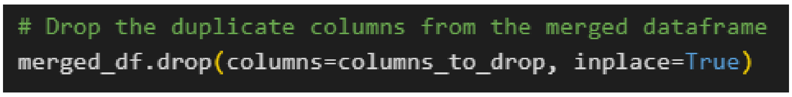
```
# Merge ALL dataframes on 'Customer_ID'
merged_df = df1.merge(df2, on='Customer_ID', how='outer') \
                .merge(df3, on='Customer_ID', how='outer') \
                .merge(df4, on='Customer_ID', how='outer') \
                .merge(df5, on='Customer_ID', how='outer')
```

3. Identifying and Dropping Duplicates

The ‘get_duplicate_columns’ function was deployed to identify any and all duplicate columns. The goal was to identify duplicates and drop them from the dataset ‘merged_df’, thus retaining only one candidate. Through this process, it was found that a significant number of columns were redundant and in need of elimination; allowing the dataset to be reduced from 102 to 40 columns. By retaining only unique columns, simplification of the dataset was possible.



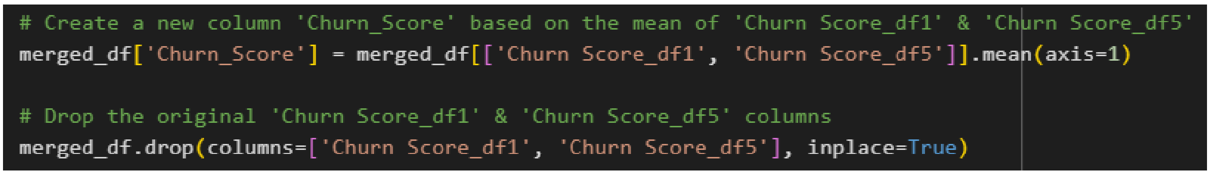
```python
# Function to identify duplicate columns
def get_duplicate_columns(df):
    duplicates = {col: compared_col for col in df.columns for compared_col in df.columns
                  if col != compared_col and df[col].equals(df[compared_col])}
    return duplicates

# Assigning duplicate columns from 'merged_df' to 'duplicate_columns'
duplicate_columns = get_duplicate_columns(merged_df)

# Create a list of columns to drop, excluding specific columns
columns_to_drop = [col for col in duplicate_columns if col not in {'Churn_df2', 'Monthly Charges_df1'}]

# Create and display a DataFrame from the list of columns to be dropped
columns_to_be_dropped = pd.DataFrame(columns_to_drop, columns=["Duplicate Column which can be dropped: "])
print(columns_to_be_dropped)
```



```python
# Drop the duplicate columns from the merged dataframe
merged_df.drop(columns=columns_to_drop, inplace=True)
```

4. Logical adjustment

Further cleansing of columns was done by comparing similarity of data in columns. Columns like ‘Churn_Score’ and ‘Churn_Reason’ were identified as needing refinement. The ‘Churn_Score’ was a numerical indicator that had been derived from previous models, while ‘Churn_Reason’ provided categorical reasons for customer churn. To ensure consistency, the ‘Churn_Score’ was adjusted by taking the mean value from related scores (‘Churn_Score_df1’ and ‘Churn_Score_df5’). This adjustment harmonized the scores, creating a single, more accurate indicator of churn likelihood. Similarly, logical adjustments were applied to ‘Churn_Reason’, ensuring that the information was consistently categorized across the dataset.

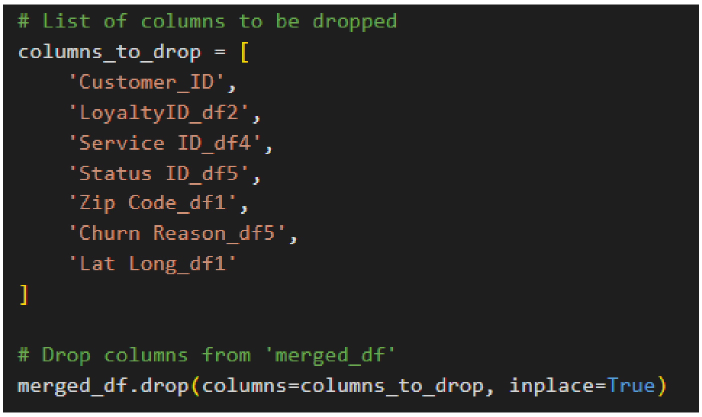


```
# Create a new column 'Churn_Score' based on the mean of 'Churn Score_df1' & 'Churn Score_df5'
merged_df['Churn_Score'] = merged_df[['Churn Score_df1', 'Churn Score_df5']].mean(axis=1)

# Drop the original 'Churn Score_df1' & 'Churn Score_df5' columns
merged_df.drop(columns=['Churn Score_df1', 'Churn Score_df5'], inplace=True)
```

5. Dropping columns

Post refinement of the dataset through logical adjustment, further cleaning was necessary and was done by dropping columns that would have been redundant for the modelling phase:

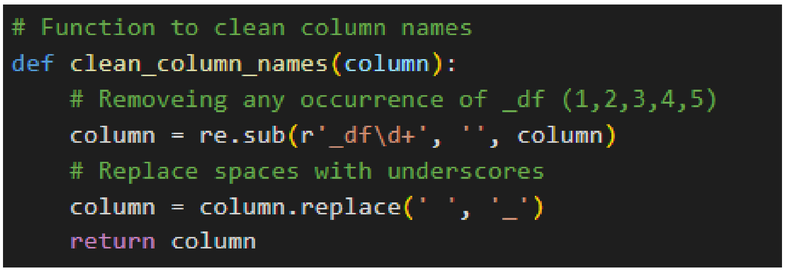
- ID's- All columns related to IDs which serve as a unique identifier was dropped as it could not contribute to the modelling but add noise and bias to it.
- Zip_Code- With over 800+ unique values, Zip Code was removed as it was too granular for meaningful analysis.
- Churn Reason_df5- This column was a duplicate of Churn Reason_df1. Since Churn Reason_df1 was derived from the main dataset, it was retained while Churn Reason_df5 was dropped.
- Lat Long_df1- Latitude and longitude data were excluded as the dataset already contained individual columns for these coordinates, which were deemed unnecessary for the analysis.



```
# List of columns to be dropped
columns_to_drop = [
    'Customer_ID',
    'LoyaltyID_df2',
    'Service ID_df4',
    'Status ID_df5',
    'Zip Code_df1',
    'Churn Reason_df5',
    'Lat Long_df1'
]

# Drop columns from 'merged_df'
merged_df.drop(columns=columns_to_drop, inplace=True)
```

6. Renaming column names

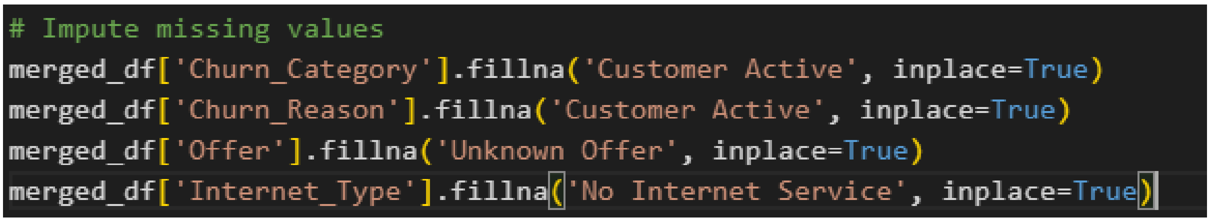
To ensure consistency and readability in the dataset, a function was implemented to clean and standardize column names. The function clean_column_names was designed to remove any occurrences of dataset-specific suffixes such as _df1, _df2, etc., which were added during the merging process. Additionally, spaces within column names were replaced with underscores to follow a uniform naming convention. This step was essential in simplifying the dataset, making it easier to reference column names in subsequent analysis and modelling tasks.



```python
# Function to clean column names
def clean_column_names(column):
    # Removeing any occurrence of _df (1,2,3,4,5)
    column = re.sub(r'_df\d+', '', column)
    # Replace spaces with underscores
    column = column.replace(' ', '_')
    return column
```

7. Handling Missing Values

- Churn_Reason: This column had 5,174 missing values, representing non-churned customers. These values were imputed with "Customer Active" to indicate that the customer was still active.
- Offer: With 3,877 missing values, this column was filled with "Unknown Offer" to account for any unrecorded offers that might have been available to the customer.
- Internet_Type: The 1,526 missing values in this column were all associated with customers who did not have an internet service. Therefore, "No Internet Service" was used as the imputed value.
- Churn_Category: Similar to Churn_Reason, this column also had 5,174 missing values for non-churned customers, which were filled with "Customer Active."



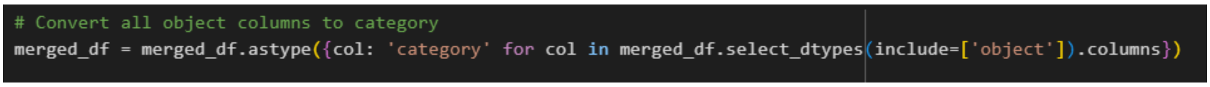
```
# Impute missing values
merged_df['Churn_Category'].fillna('Customer Active', inplace=True)
merged_df['Churn_Reason'].fillna('Customer Active', inplace=True)
merged_df['Offer'].fillna('Unknown Offer', inplace=True)
merged_df['Internet_Type'].fillna('No Internet Service', inplace=True)
```

8. Data Type Conversions

To further enhance the efficiency of data processing and improving performance of the predictive models, it was best to convert several variables into categorical data types. This conversion simplifies analysis and makes the data more suitable for modelling.

The following variables were re-classified:

- Dependents, Streaming Movies, Tech Support, Phone Service, Senior Citizen, Gender, Online Backup, Streaming Music, Multiple Lines, Streaming TV, Device Protection, Unlimited Data, Partner, Paperless Billing, Online Security and Churn: Originally represented as object or integer types, these variables were converted to categorical data types, with values such as "Yes" or "No."
- Internet Service, Contract, and Payment Method: These variables, which describe the type of service or customer preferences, were also converted to categorical types, with multiple levels like "DSL," "Fiber optic," "Month-to-month," etc.



```
# Convert all object columns to category
merged_df = merged_df.astype({col: 'category' for col in merged_df.select_dtypes(include=['object']).columns})
```

9. Outlier Detection

Outliers within the dataset were initially identified using the Z-score method. This approach involved calculating the Z-scores for each numeric column, which represents how many standard deviations an element is from the mean. A threshold of 3 was used to flag any data points as outliers if their absolute Z-score exceeded this value.

Outlier Counts before Capping

The Z-score method revealed the following counts of outliers across various columns:

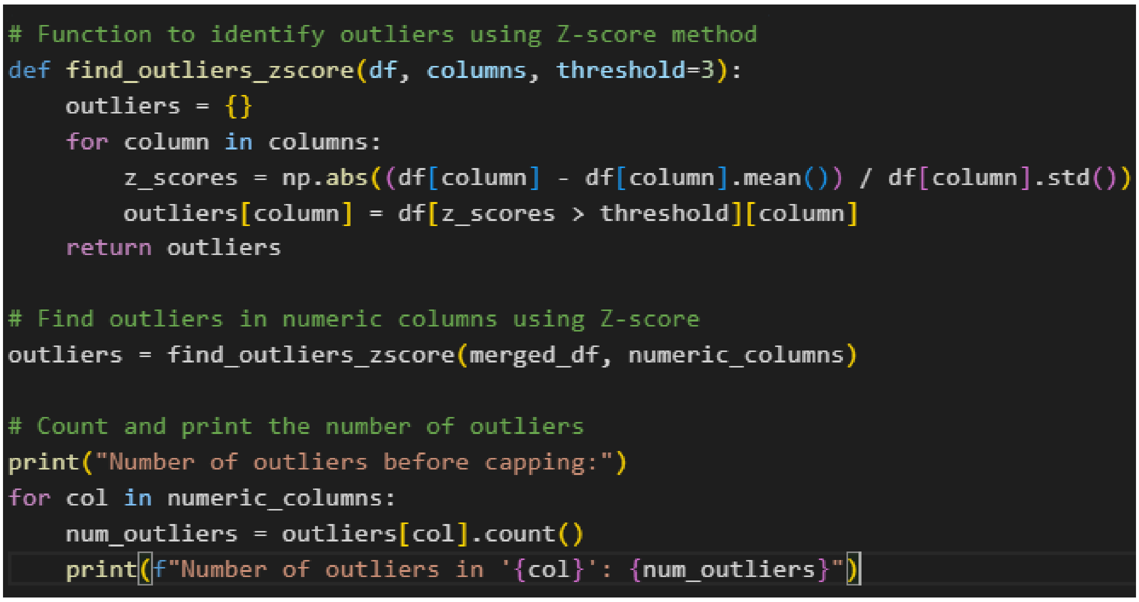
- Number_of_Dependents: 26 outliers
- Number_of_Referrals: 2 outliers
- Avg_Monthly_GB_Download: 91 outliers
- Total_Refunds: 266 outliers
- Total_Extra_Data_Charges: 259 outliers
- Total_Long_Distance_Charges: 58 outliers
- Total_Revenue: 5 outliers

Other columns showed no significant outliers.

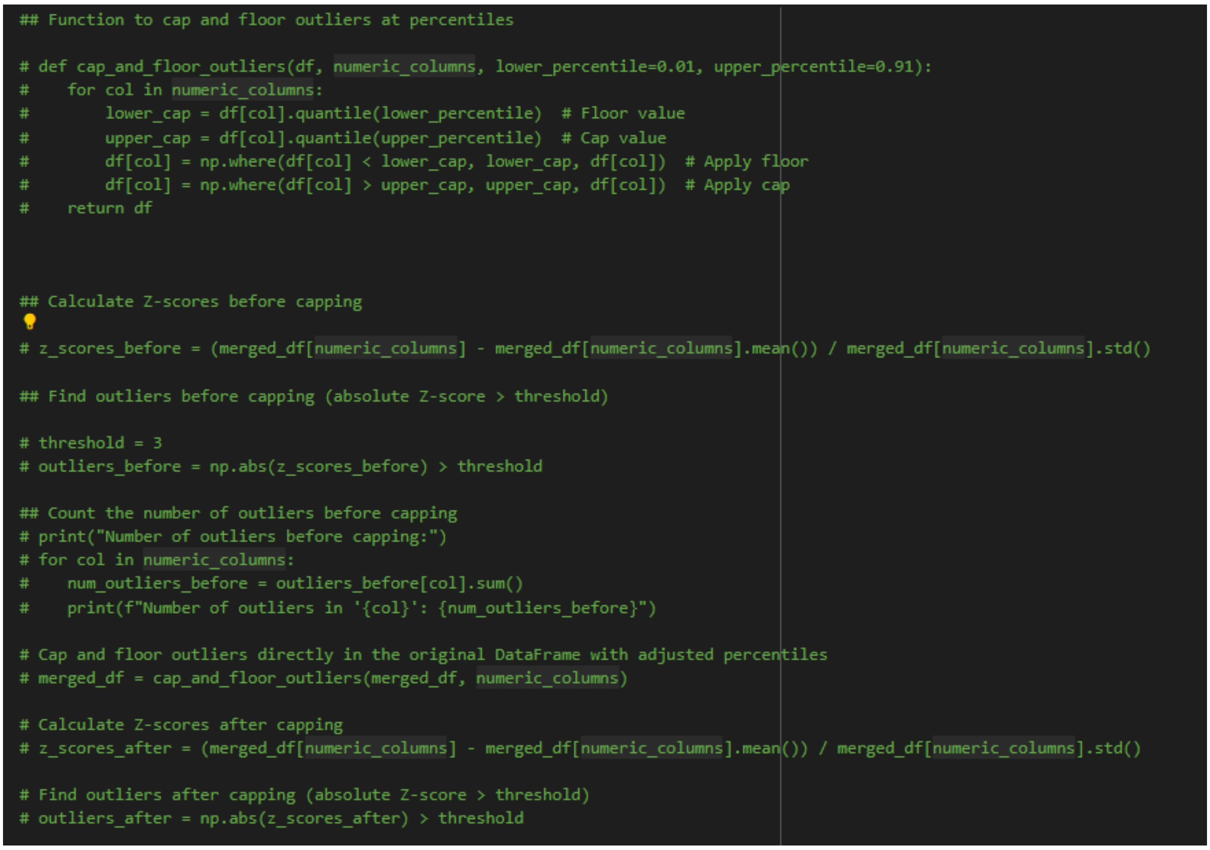
Capping and Flooring Outliers

An attempt was made to mitigate the impact of these outliers by capping and flooring values at the 1st and 91st percentiles, respectively. However, this approach led to skewed data and introduced imbalances that could negatively impact the model's performance.

Due to these challenges, the decision was made to carefully reconsider the handling of outliers to ensure the data remained balanced and representative of the underlying patterns without artificially distorting the dataset.



```python
# Function to identify outliers using Z-score method
def find_outliers_zscore(df, columns, threshold=3):
    outliers = {}
    for column in columns:
        z_scores = np.abs((df[column] - df[column].mean()) / df[column].std())
        outliers[column] = df[z_scores > threshold][column]
    return outliers

# Find outliers in numeric columns using Z-score
outliers = find_outliers_zscore(merged_df, numeric_columns)

# Count and print the number of outliers
print("Number of outliers before capping:")
for col in numeric_columns:
    num_outliers = outliers[col].count()
    print(f"Number of outliers in '{col}': {num_outliers}")
```



```python
## Function to cap and floor outliers at percentiles

# def cap_and_floor_outliers(df, numeric_columns, lower_percentile=0.01, upper_percentile=0.91):
#    for col in numeric_columns:
#        lower_cap = df[col].quantile(lower_percentile)  # Floor value
#        upper_cap = df[col].quantile(upper_percentile)  # Cap value
#        df[col] = np.where(df[col] < lower_cap, lower_cap, df[col])  # Apply floor
#        df[col] = np.where(df[col] > upper_cap, upper_cap, df[col])  # Apply cap
#    return df



## Calculate Z-scores before capping

# z_scores_before = (merged_df[numeric_columns] - merged_df[numeric_columns].mean()) / merged_df[numeric_columns].std()

## Find outliers before capping (absolute Z-score > threshold)

# threshold = 3
# outliers_before = np.abs(z_scores_before) > threshold

## Count the number of outliers before capping
# print("Number of outliers before capping:")
# for col in numeric_columns:
#    num_outliers_before = outliers_before[col].sum()
#    print(f"Number of outliers in '{col}': {num_outliers_before}")

# Cap and floor outliers directly in the original DataFrame with adjusted percentiles
# merged_df = cap_and_floor_outliers(merged_df, numeric_columns)

# Calculate Z-scores after capping
# z_scores_after = (merged_df[numeric_columns] - merged_df[numeric_columns].mean()) / merged_df[numeric_columns].std()

# Find outliers after capping (absolute Z-score > threshold)
# outliers_after = np.abs(z_scores_after) > threshold
```

Data Dictionary

Variable	Data Type	Values
Country	category	['United States']
State	category	['California']
City	category	[too many to list]
Latitude	float64	Numerical
Longitude	float64	Numerical
Monthly_Charges	float64	Numerical
Churn_Reason	category	['Competitor made better offer' 'Moved' 'Competitor had better devices' 'Competitor offered higher download speeds' 'Competitor offered more data' 'Price too high' 'Product dissatisfaction' 'Service dissatisfaction' 'Network reliability']
Dependents	category	['No' 'Yes']
Churn	category	['Yes' 'No']
Age	int64	Numerical

Under_30	category	['No' 'Yes']
Number_of_Dependents	int64	Numerical
Referred_a_Friend	category	['No' 'Yes']
Number_of_Referrals	int64	Numerical
Tenure_in_Months	int64	Numerical
Offer	category	['Unknown Offer' 'Offer C' 'Offer E' 'Offer D' 'Offer A' 'Offer B']
Avg_Monthly_Long_Distance_Charges	float64	Numerical
Multiple_Lines	category	['No' 'Yes']
Internet_Service	category	['Yes' 'No']
Internet_Type	category	['DSL' 'Fiber Optic' 'Cable' 'No Internet Service']
Avg_Monthly_GB_Download	int64	Numerical
Online_Security	category	['Yes' 'No']
Online_Backup	category	['Yes' 'No']
Device_Protection_Plan	category	['No' 'Yes']
Premium_Tech_Support	category	['No' 'Yes']
Streaming_TV	category	['No' 'Yes']
Streaming_Movies	category	['No' 'Yes']
Streaming_Music	category	['No' 'Yes']
Unlimited_Data	category	['Yes' 'No']
Contract	category	['Month-to-Month' 'Two Year' 'One Year']

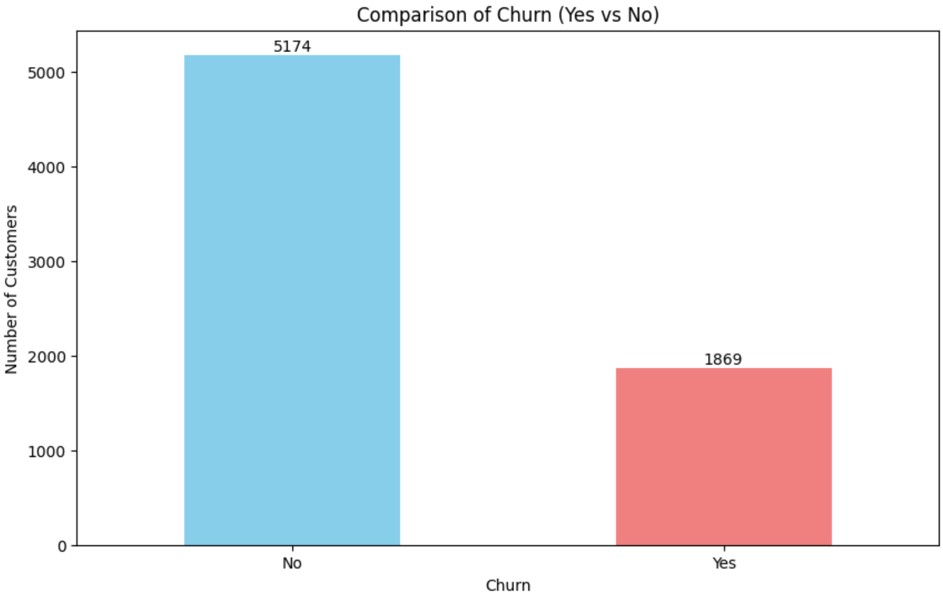
Payment_Method	category	['Credit Card' 'Bank Withdrawal' 'Mailed Check']
Total_Charges	float64	Numerical
Total_Refunds	float64	Numerical
Total_Extra_Data_Charges	int64	Numerical
Total_Long_Distance_Charges	float64	Numerical
Total_Revenue	float64	Numerical
Satisfaction_Score	int64	Numerical
Customer_Status	category	['Churned' 'Joined' 'Stayed']
Churn_Category	category	['Competitor' 'Other' 'Price' 'Dissatisfaction' 'Attitude' 'Customer Active']
Churn_Score	float64	Numerical

Data Exploration

For the cleaned dataset, data exploration is a vital step in understanding and visualizing key data points that help us understand the data better. Data exploration or Exploratory Data Analysis (E.D.A) allows us to investigate customer behaviours, churn reasons, patterns in customer characteristics and much more that influence churn. Through this process, we can notice any anomalies, outliers (which is where I found my cap & floor to be yielding skewed data) and critical insights into the data.

Customer Churn Distribution

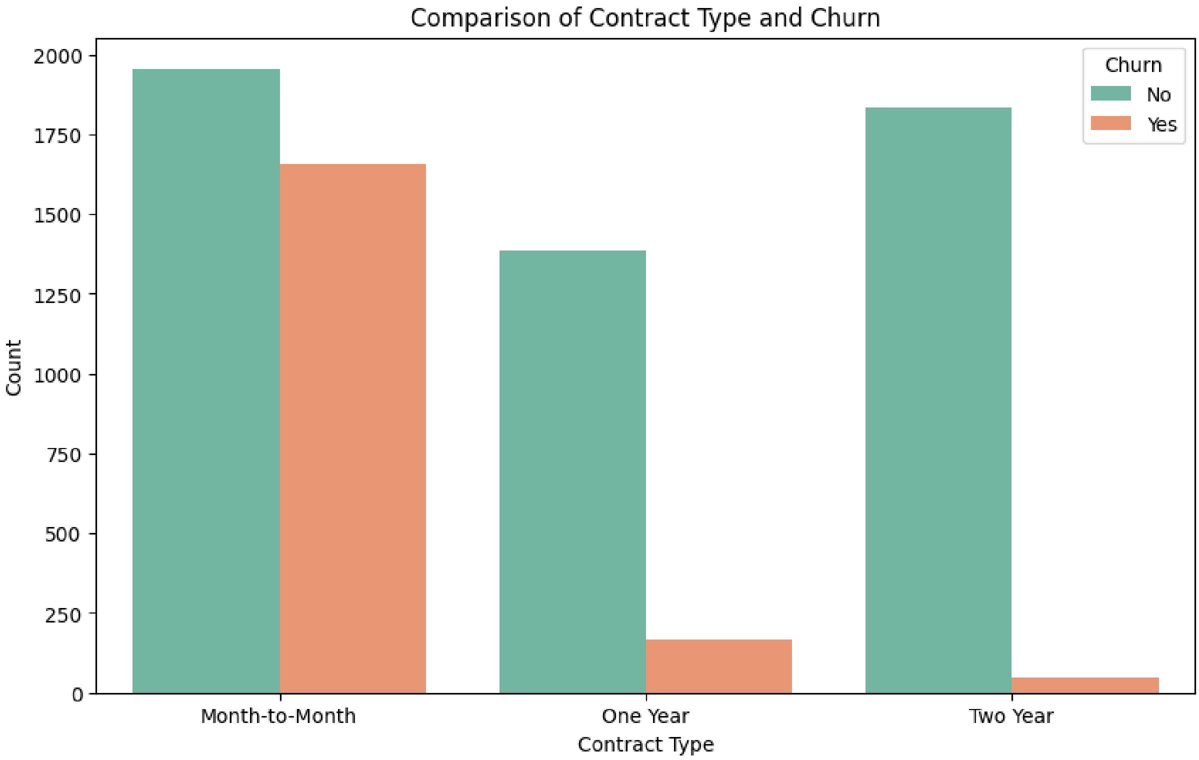
The bar chart provides a compelling visualization of customer churn at Vortex Telecommunications. Out of the total customer base, 5,174 customers have remained loyal, while 1,869 customers have churned. This was determined by counting the occurrences of 'Yes' and 'No' in the Churn column.



Comparison of Contract Type and Churn

The histogram illustrates the relationship between contract types of customers and customer churn rates at Vortex. By comparing the churn rates across different contract lengths we notice trends emerging.

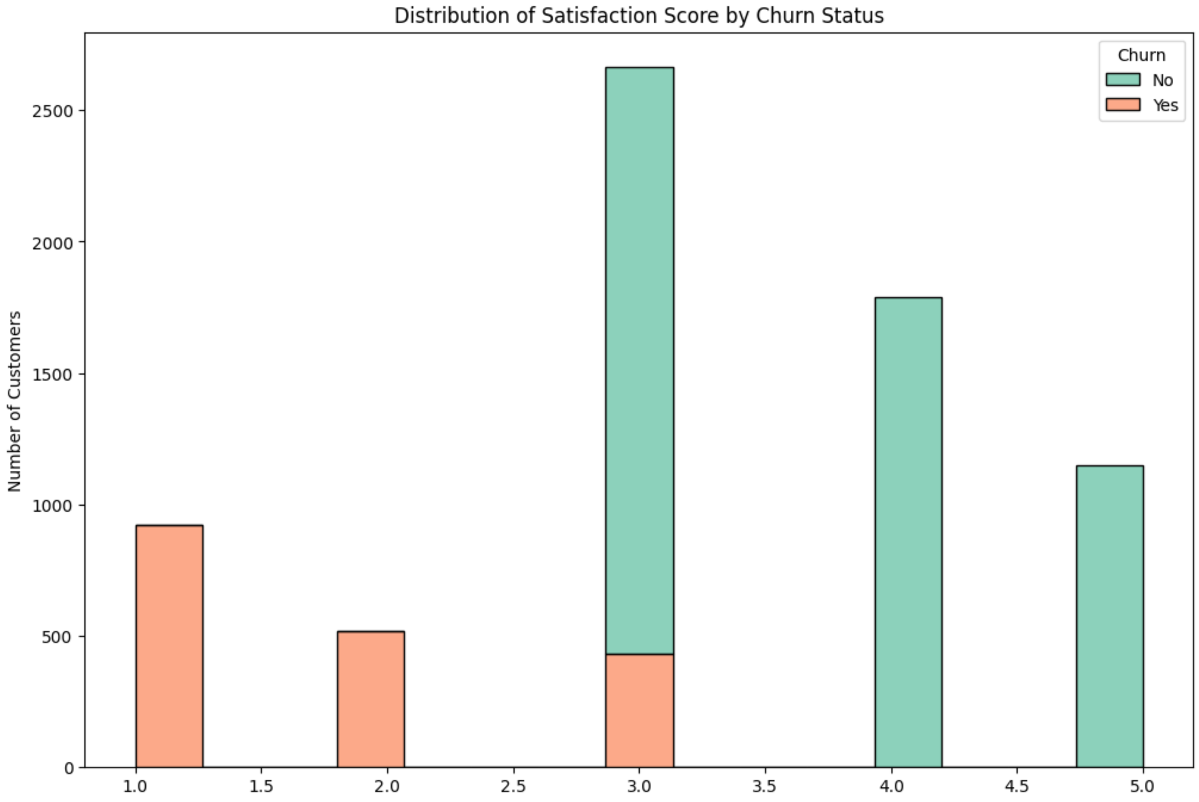
- Month-to-Month Contracts: Customers on month-to-month contracts show the highest churn rate, with a relatively balanced count between those who stayed and those who churned.
- One-Year Contracts: The churn rate significantly drops for customers on one-year contracts, with a much higher count of customers staying compared to those who churned.
- Two-Year Contracts: The churn rate is the lowest for customers on two-year contracts, indicating higher customer retention for longer-term commitments.



Distribution of Satisfaction Score by Churn Status

The below histogram provides an understanding of how customer satisfaction scores directly correlates with churn rates. By viewing the distribution of satisfaction scores among customers who have churned vs those who have stayed, clear patterns emerge.

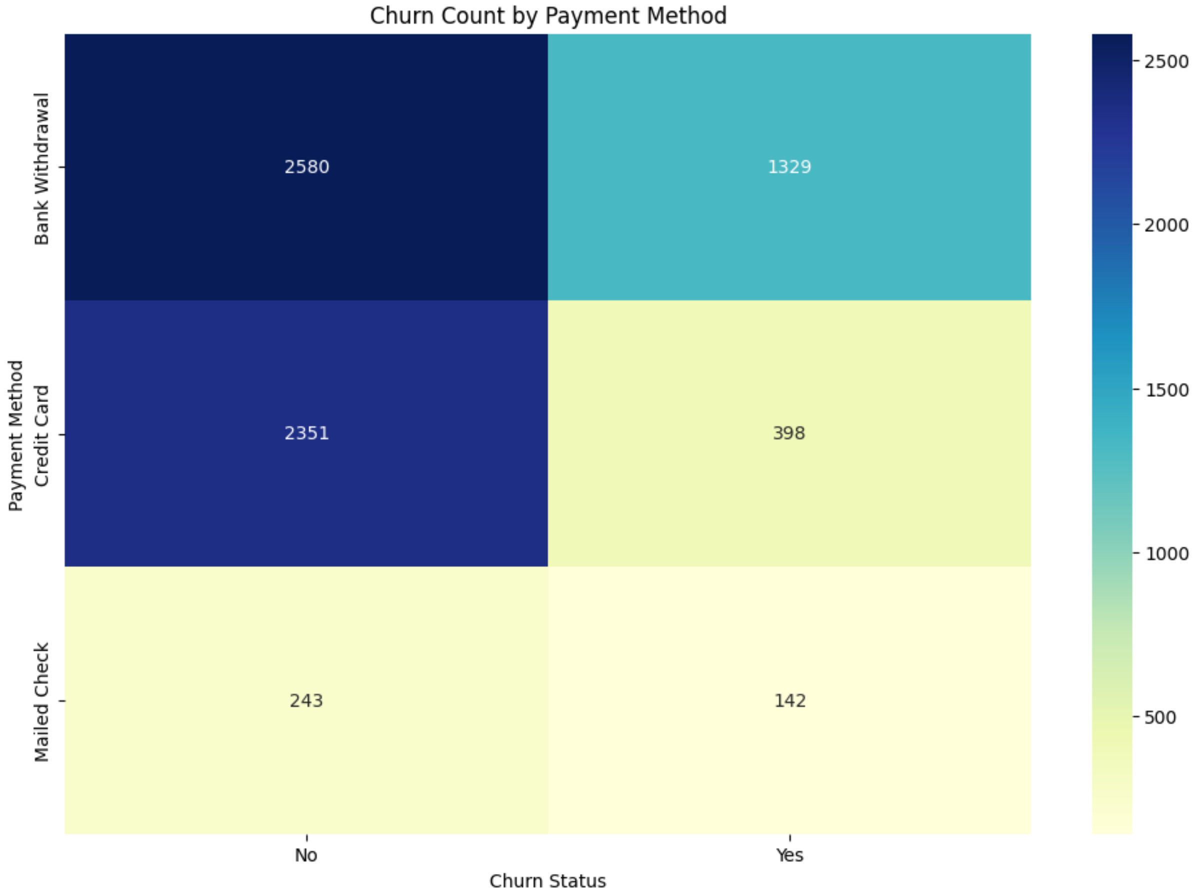
- Low Satisfaction Scores (1.0 - 2.0): Customers with lower satisfaction scores show a significantly higher likelihood of churn. The majority of churned customers fall within this range.
- Moderate Satisfaction Scores (3.0): The bulk of customers in this score range have not churned, indicating a more stable customer base with moderate satisfaction.
- High Satisfaction Scores (4.0 - 5.0): Customers with higher satisfaction scores are predominantly loyal, with very few instances of churn observed in these groups.



Churn Count by Payment Method

The below heat map visualizes the relationship between various payment methods used by customers and churns. By examining the churn counts across the various payment methods, we notice patterns that indicate on customer preferences and behaviours.

- Bank Withdrawal: This payment method has the highest customer count, with 2,580 customers not churning and 1,329 customers who have churned. The high churn count suggests that customers using bank withdrawal may require targeted retention efforts.
- Credit Card: A total of 2,351 customers using credit cards have not churned, while 398 have. The lower churn rate compared to bank withdrawal indicates potentially higher customer satisfaction among credit card users.
- Mailed Check: This method has the fewest customers overall, with 243 remaining and 142 churning. The relatively balanced churn ratio might suggest a specific customer segment that prefers traditional payment methods.

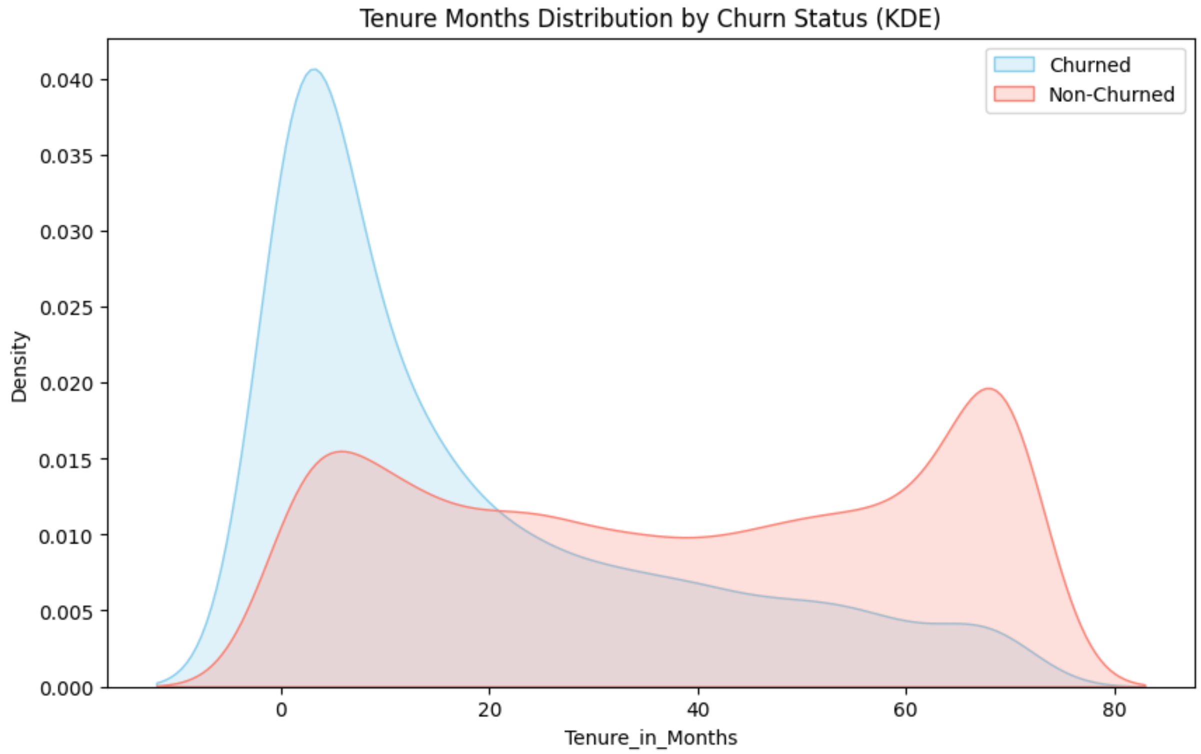


Tenure Months Distribution by Churn Status (KDE)

The below kernel density estimate plot shows us the distribution of customer tenure in months to churns yes or no.

- Churned Customers: The KDE plot shows a significant peak for churned customers at very low tenure (around 0-10 months), indicating that customers with short tenures are more likely to churn.
- Non-Churned Customers: Non-churned customers demonstrate a more varied distribution, with a noticeable peak at around 60-70 months, suggesting that longer-tenured customers are more likely to remain loyal.
- Overlap and Differences: There is an overlap in the mid-range tenures (20-40 months), but the sharp differences at the extremes highlight the importance of customer tenure as a predictor of churn.

This plot underscores the critical role of customer tenure in predicting churn, with shorter tenures being a strong indicator of potential churn risk.

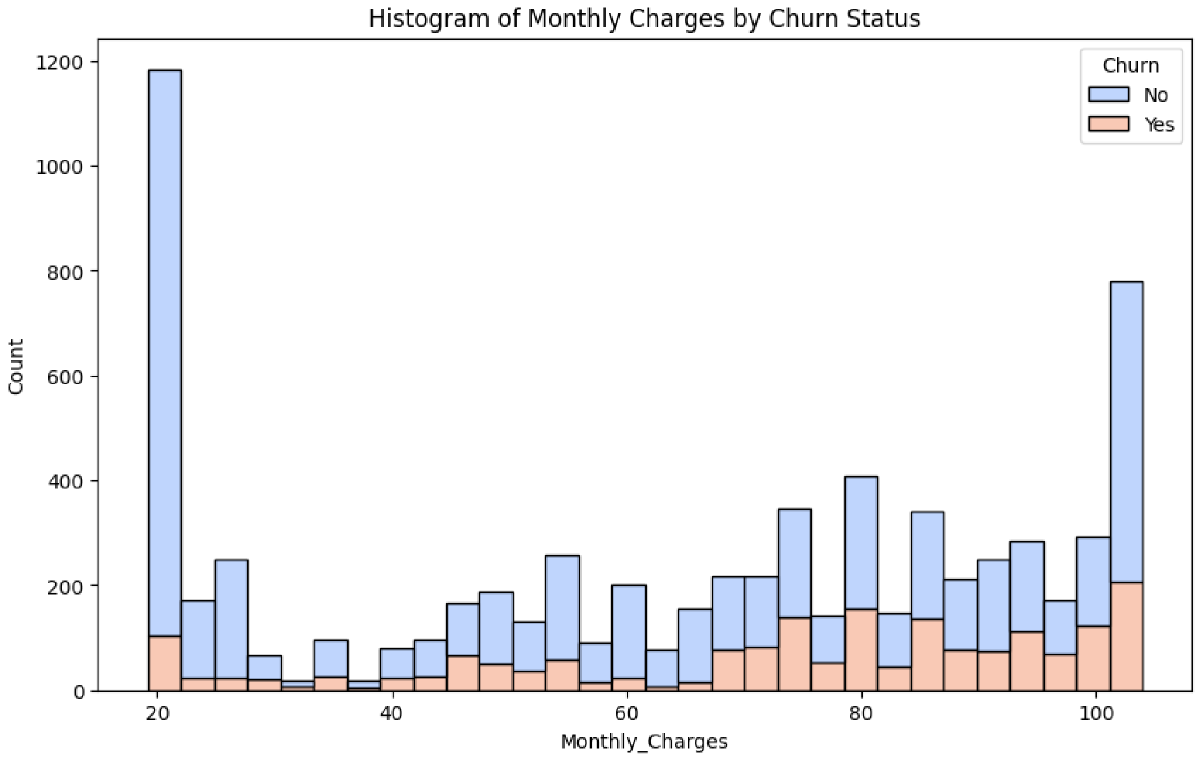


Monthly Charges by Churn Status

Below histogram visualizes distribution of monthly charges paid by customers across churned and non-churned.

- Low Monthly Charges (~$20): The highest concentration of customers falls within the low monthly charges range, with a significant portion of non-churned customers, though a noticeable number of churned customers also exist in this range.
- Moderate Monthly Charges ($40 - $80): As monthly charges increase, there is a balanced mix of churned and non-churned customers, indicating that churn occurs consistently across these charge levels.
- High Monthly Charges (~$100): At the higher end of monthly charges, the churn rate slightly increases again, but non-churned customers still dominate this range.

This distribution suggests that churn is not isolated to any specific range of monthly charges but occurs across the board, with notable peaks at both the low and high ends of the spectrum.

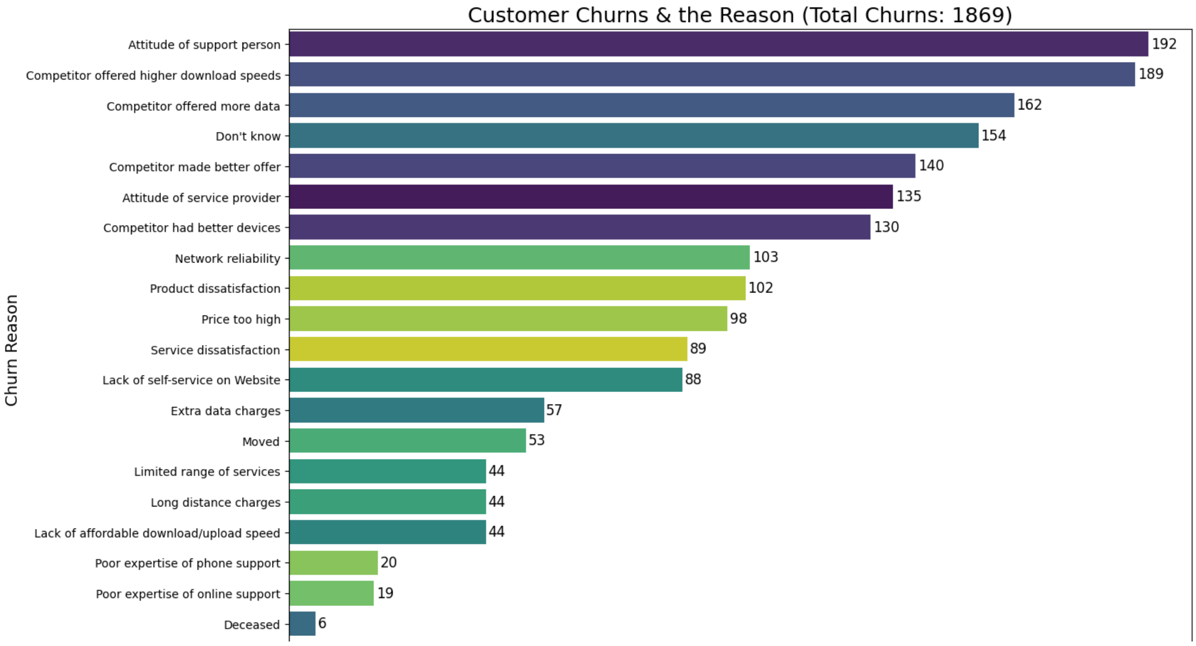


Customer Churns & the Reason

This bar chart provides a detailed breakdown of the reasons behind customer churn at Vortex Telecommunications, showcasing the frequency of each reason among the 1,869 churned customers.

- Top Reasons for Churn: The most common reasons for churn include dissatisfaction with the attitude of support personnel (192 occurrences) and competitors offering higher download speeds (189 occurrences).
- Competitor Influence: Several churn reasons are directly related to competitor offerings, such as better data plans, devices, and overall offers, highlighting the competitive pressure faced by Vortex.
- Service and Product Dissatisfaction: Other significant factors include network reliability issues, product dissatisfaction, and high prices, each contributing to over 100 churn cases.
- Minor Factors: Less frequent reasons for churn include poor expertise in phone support and online support, as well as customers moving away or being deceased.

This visualization underscores the need for Vortex to address customer service issues and improve their offerings to stay competitive and reduce churn rates.



Heatmap of Numeric Columns

The below Heatmap provides a visual representation of the correlations between various numeric columns in the dataset. The correlation coefficient between two variables is represented by each cell in the Heatmap, and its values range from -1 to 1. A correlation close to 1 indicates a strong positive relationship, and a correlation close to -1 indicates a strong negative relationship. A correlation close to zero indicates little to no linear relationship.

- Monthly Charges:
  - Total Charges (0.65): Monthly Charges are moderately positively correlated with Total Charges, which is expected since higher monthly charges contribute to higher total charges over time.
  - Avg Monthly GB Download (0.39): There is moderate positive correlation, suggesting that customers who incur a higher monthly charges use more data.
  - Total Revenue (0.59): A moderate positive correlation indicates that higher monthly charges contribute to higher total revenue.
  - Churn Score (0.13): A slight positive correlation suggests that higher monthly charges may slightly increase the likelihood of churn.

- Age:
  - Monthly Charges (0.14): There is a weak positive correlation, indicating that age slightly influences monthly charges, with older customers possibly paying slightly more.
  - Satisfaction Score (-0.085): A weak negative correlation suggests that satisfaction slightly decreases with age.
  - Churn Score (0.084): A weak positive correlation with churn score indicates that older customers might be slightly more likely to churn.

- Number of Dependents:
  - Number of Referrals (0.28): A moderate positive correlation suggests that customers with more dependents tend to refer others more often.

  - Tenure in Months (0.11): A weak positive correlation suggests that customers with dependents tend to stay longer with the service.
  - Churn Score (-0.16): A moderate negative correlation indicates that customers with dependents are less likely to churn.

- Number of Referrals:
  - Tenure in Months (0.33): A moderate positive correlation suggests that customers who stay longer tend to refer more.
  - Total Revenue (0.18): A weak positive correlation indicates that more referrals are associated with slightly higher revenue.

- Tenure in Months:
  - Total Charges (0.83): A strong positive correlation indicates that longer-tenured customers have accumulated higher charges over time.
  - Total Revenue (0.85): A strong positive correlation shows that longer-tenured customers contribute significantly to the total revenue.
  - Churn Score (-0.23): A moderate negative correlation suggests that longer tenure is associated with a lower likelihood of churn.

- Avg Monthly Long Distance Charges:
  - Total Long Distance Charges (0.67): A strong positive correlation indicates that higher average monthly long-distance charges lead to higher total long-distance charges.
  - Churn Score (0.022): A very weak positive correlation indicates that higher average long-distance charges have minimal impact on churn.

- Avg Monthly GB Download:
  - Monthly Charges (0.39): As noted earlier, more data usage is associated with higher monthly charges.
  - Total Charges (0.25): A moderate positive correlation shows that higher data usage contributes to higher total charges.

  - Satisfaction Score (-0.092): A weak negative correlation suggests that higher data usage might slightly decrease satisfaction.

- Total Charges:
  - Total Revenue (0.97): This nearly perfect correlation is expected, as total charges directly contribute to total revenue.
  - Total Long Distance Charges (0.78): A strong positive correlation indicates that customers with higher total charges also tend to have higher long-distance charges.
  - Satisfaction Score (0.095): A very weak positive correlation indicates that higher total charges have a minimal positive impact on satisfaction.
  - Churn Score (-0.14): A weak negative correlation suggests that higher total charges slightly decrease the likelihood of churn.

- Total Refunds:
  - Satisfaction Score (-0.024): A very weak negative correlation suggests that refunds slightly decrease satisfaction, which is intuitive.
  - Churn Score (0.028): A very weak positive correlation indicates that customers who received refunds might be slightly more likely to churn.

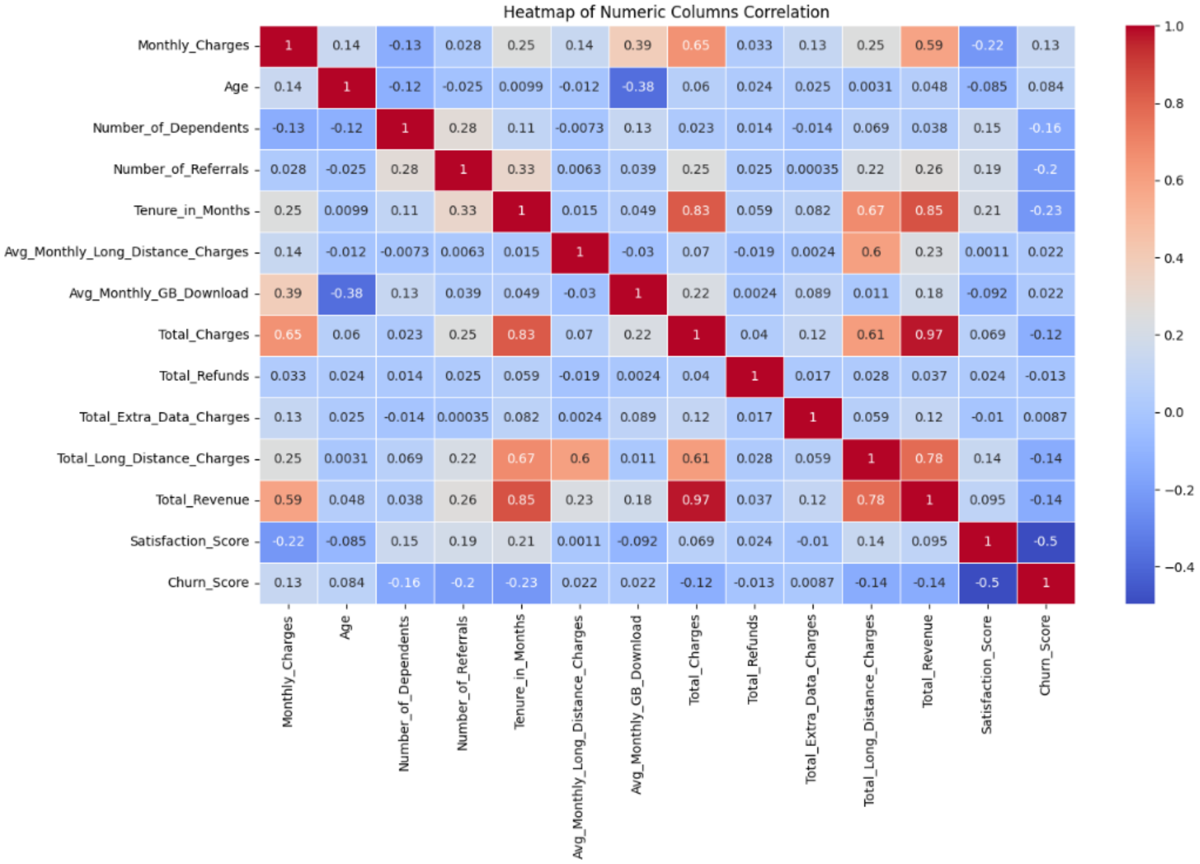
- Total Extra Data Charges:
  - Total Charges (0.12): A weak positive correlation shows that extra data charges contribute to higher total charges.
  - Total Revenue (0.12): Similarly, extra data charges slightly increase total revenue.
  - Churn Score (-0.007): A near-zero correlation suggests that extra data charges have almost no impact on churn.

- Total Long Distance Charges:
  - Total Revenue (0.78): A strong positive correlation indicates that long-distance charges significantly contribute to total revenue.
  - Churn Score (-0.14): A weak negative correlation suggests that higher long-distance charges slightly reduce the likelihood of churn.

- Total Revenue:
  - Satisfaction Score (0.095): A very weak positive correlation indicates that higher revenue is slightly associated with higher satisfaction.
  - Churn Score (-0.14): A weak negative correlation suggests that higher revenue might slightly decrease the likelihood of churn.

- Satisfaction Score:
  - Churn Score (-0.5): A moderate negative correlation shows that higher satisfaction scores are associated with significantly lower churn scores, confirming that satisfied customers are less likely to churn.

- Churn Score:
  - Various Factors: The Churn Score is moderately negatively correlated with satisfaction and weakly correlated with several other factors, reinforcing its role as an indicator of the likelihood of customer churn.



Summary of Descriptive Statistics

- Monthly Charges: Customers tend to pay an average of $64.33 per month, with monthly charges ranging from $19.20 to $103.95. This wide range reflects the diverse service plans and usage patterns among the customer base.
- Age: The average age of customers is 46.15 years, with ages spanning from 19 to 72 years old. This suggests a broad demographic spread among Vortex's customers, encompassing both younger and older age groups.
- Number of Dependents: On average, customers have 0.38 dependents, with the number of dependents ranging from 0 to 2. This indicates that most customers have few or no dependents.

- Number of Referrals: Customers have made an average of 2 referrals, with the number of referrals ranging from 0 to 8. This suggests some customers are actively referring others to Vortex's services.
- Tenure in Months: The average customer tenure is 32.26 months, varying between 1 and 70 months. This indicates a substantial proportion of long-term customers, although some are relatively new to the service.
- Avg Monthly Long Distance Charges: Customers incur an average of $22.73 in long-distance charges monthly, with charges ranging from $0.00 to $45.04. This shows that long-distance service usage varies considerably among customers.
- Avg Monthly GB Download: Customers download an average of 19.43 GB per month, with data usage ranging from 0 to 56 GB. This suggests a significant variation in data consumption across the customer base.
- Total Charges: The average total charges for customers are $2,191.47, with amounts ranging from $19.90 to $6,141.03. This wide range reflects the cumulative impact of varied service plans and usage over time.
- Total Refunds: The average total refunds are $0.00, indicating no significant refunds were recorded in the dataset.
- Total Extra Data Charges: Customers incurred an average of $1.03 in extra data charges, with a range from $0.00 to $10.00. This indicates that extra data charges are generally minimal but can be significant for some customers.
- Total Long Distance Charges: The average total long distance charges are $702.13, with charges ranging from $0.00 to $2,181.09. This highlights the variability in long-distance usage among the customer base.
- Total Revenue: Customers contribute an average total revenue of $2,919.06, with contributions ranging from $31.02 to $7,840.43. This suggests that customer revenue varies widely depending on service usage and tenure.
- Satisfaction Score: The average customer satisfaction score is 3.24(0-5 scale), indicating a generally moderate level of customer satisfaction.
- Churn Score: The average churn score is 58.17, with scores ranging from 20 to 88. This metric indicates the likelihood of customer churn, with higher scores representing a greater risk of churn.

Pre-Modelling Data Readiness

Following the data pre-processing steps above, the next phase in the project is to get the dataset further ready for modelling. This is where pre-modelling data readiness comes in, where we address tasks like creation of dummy variables, creation of test & train datasets and so on. Below are all of the steps taken to get the data ready for modelling.

Further Cleaning

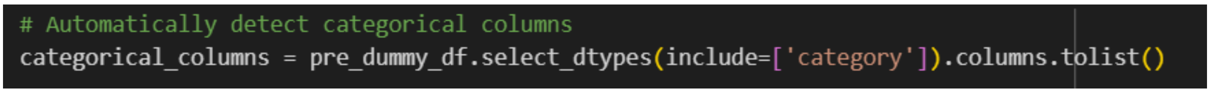
The reason why we didn’t initially clean all data was to visualize them in our data visualization steps above. Now that the visualizations are completed, we can further clean data to ensure accurate prediction of our models. There are still several variables that may cause bias in modelling due to them being irrelevant and their granularity. Below are the columns that were further dropped:

Variable	Description
Country	Just 1 country in the dataset.
State	Just 1 state in the dataset.
City	850+ Cities, which skews the modelling.
Latitude	Extremely granular, and unnecessary for modelling.
Longitude	Extremely granular, and unnecessary for modelling.
Churn_Score	This is the score derived from a previous modelling effort, from where the dataset was sourced.
Churn_Reason	Same as above.
Churn_Category	Same as above.
Satisfaction_Score	Makes modelling too easy, as the model accurately predicts churn based off obvious satisfaction scores.
Total_Long_Distance_Charges	Variables act as bias in modelling; charges are obvious indicators of churn.
Total_Charges	
Total_Extra_Data_Charges	
Total_Revenue	
Total_Monthly_Charge	
Total_Long_Distance_Charge	

Creation of Dummy Variables

For predictive modelling, a critical task is to transform all categorical variables into a readable format for all the machine learning models for better interpretability. This transformation of the data can be done by creating dummy variables of all the categorical variables.

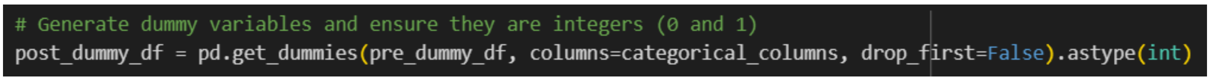
First, we detect all of the categorical columns in ‘pre_dummy_df’. Categorical variables are those that represent distinct categories or labels rather than continuous numerical values. In this project, the categorical columns were automatically detected using the select_dtypes function, which filters the dataset to include only those columns with a 'category' data type.



```
# Automatically detect categorical columns
categorical_columns = pre_dummy_df.select_dtypes(include=['category']).columns.tolist()
```

This code identifies all columns in ‘pre_dummy_df’ that are categorized as 'category' types and stores their names in the ‘categorical_columns’ list. This step is crucial because it ensures that only relevant categorical data undergoes the transformation process.

Once all categorical columns have been identified, the next process is to convert these columns into dummy variables. Using ‘pd.get_dummies’ this transformation can be achieved.



```
# Generate dummy variables and ensure they are integers (0 and 1)
post_dummy_df = pd.get_dummies(pre_dummy_df, columns=categorical_columns, drop_first=False).astype(int)
```

An important parameter to set is ‘drop_first=False’. This ensures that all parameters are retained. Whereas if =True, the first parameter is dropped from the dataset. False ensures that if there are columns with more than 2 parameters, there is no miss of any parameters.

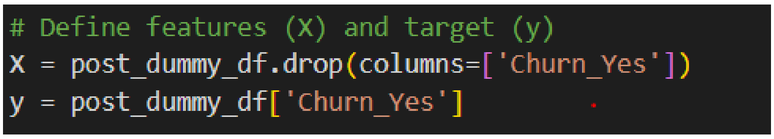
By creating dummy variables, the dataset is transformed into a format that is both machine-readable and capable of being processed by predictive models.

Splitting of Data into Training & Testing Sets

The dataset is divided into training and testing sets as the final step in data readiness. This is an important step because it allows the following models to be trained on a subset of the data while saving a set of data for testing purposes, ensuring that the model has not seen that data. The goal is to ensure that machine learning models can accurately generalize to new data.

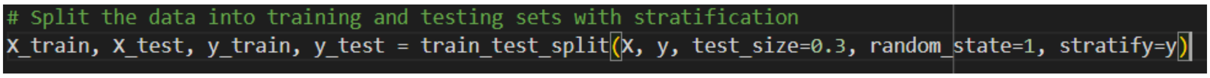
Defining Features and Target Variable

Before splitting the dataset into train & test, we need to define X (features) and y (target variable). Our target variable is ‘Churn_Yes’ as we are predicting for all churns in the dataset. The remaining columns in the dataset post dropping of target variable is to be considered as X.



```
# Define features (X) and target (y)
X = post_dummy_df.drop(columns=['Churn_Yes'])
y = post_dummy_df['Churn_Yes']
```

Once X & y are defined, we use ‘train_test_split’ from sklearn.model_selection library to split the dataset. After multiple iterations, a 70-30 split was deemed best for modelling. Which means that 70% of the data will be used to train the model and the remaining 30% of the data will be used to test the model. Additionally, stratification is applied to ensure the proportion of churn vs non-churn customers remain constant across both sets of data. This is crucial and prevents any imbalance in data that could potentially skew the model.



```
# Split the data into training and testing sets with stratification
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1, stratify=y)
```

By splitting the dataset as seen above, the model is trained & tested on well balanced data to provide accurate modelling results. This careful division of data helps ensure that the model performs well in real-world scenarios, leading to actionable insights for Vortex Telecommunications.

Model Exploration

To predict customer churn for the dataset we have on hand, a variety of machine learning models could be used. The models used in this project are:

- Logistic Regression (full, forward & backward): Being a fundamental model that is used for binary classifications similar to churn predictions, in this project full, forward and backward regressions were implemented to predict the likelihood of churn. Full logistic regression uses all available features to predict churn. Forward regression starts off with no features being selected and incrementally adds on features, while backward regression starts off with all available features and proceeds with removing least significant features.
- K Nearest Neighbour: K-Nearest Neighbours (KNN) is a non-parametric method utilized for classification tasks (Patel & Upadhyay, 2014, p. 53). It classifies customers based on their similarity of features to the nearest data point in the training set. KNN is excellent in capturing relationships between variables.
- Neural Network using MLPClassifier: Using the sklearn.neural_network module, the Multi-Layer Perceptron (MLP) Classifier is a powerful tool for capturing complex patterns in datasets (GaborMelli, n.d.). The model is used to predict customer churns by learning the dataset and making accurate predictions.
- Random Forest: The Random Forest model, being an ensemble learning approach that constructs multiple decision trees, is particularly effective in handling large datasets and mitigating overfitting (Pradeep & Kumar, 2023, p. 4376). Random Forest provides a strong predictive model for us by considering multiple predictions of the decision trees and aggregating them to make an accurate and stable prediction. Random Forest is extremely useful when the dataset has a mix of categorical and numerical features, similar to what we have on hand.

Each of these models was applied to the Vortex Telecommunications dataset, with their performances evaluated and compared to identify the most effective approach for predicting customer churn.

Metrics

There are three key metrics that were used to evaluate the models that predict customer churns. Receiver Operating Characteristic - Area Under the Curve (ROC-AUC), Accuracy, and F1-score.

- ROC-AUC- ROC-AUC metric evaluates a machine learning model's ability to differentiate between customers who are likely to churn and those who are active or likely to stay with Vortex. A higher ROC score indicates that the model is reliable and accurate in predicting customer behaviors (FasterCapital, n.d.).
- Accuracy- Accuracy measure the proportion of accurate predictions for both churns as well as non-churns based out of all predictions made. The higher the accuracy score, the overall effectiveness of a model in predicting which customer churns and which do not.
- F1-Score- F1-score is a combination of precision (how accurate the model's positive predictions are) and recall (its ability to capture actual churn cases). A higher F1 score indicates that the model is good at correctly identifying churn cases as churns and non-churn cases as non-churns. This metric is particularly important for evaluating how well the model handles the class of customers who are predicted to churn, minimizing false positives.

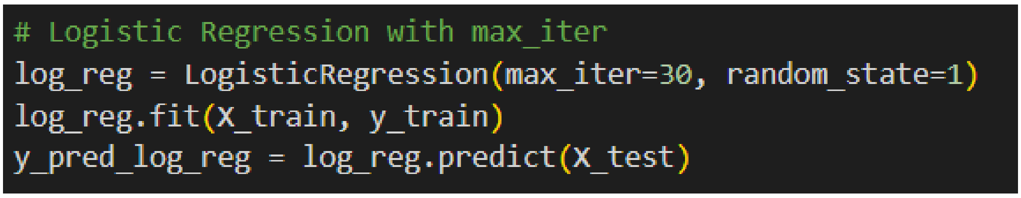
Logistic Regression

Logistic regression being a popular classification algorithm for binary classification tasks, suited the need to predict churn probabilities of Vortex. Logistic regression estimates the likelihood of customer churn based on various customer features. The model outputs a probability between 0 and 1, which can be interpreted as the likelihood of churn. If the probability exceeds a certain threshold, the customer is classified as likely to churn.

Logistic Regression- Model Training

Training of the logistic regression model was done by using 'LogisticRegression' function from the sklearn.linear_model library. The model's training process involved fitting it to the training data (X_train and y_train), allowing it to learn the relationships between the customer features and the likelihood of churn.

- The model was configured with max_iter=30 to control the number of iterations during the optimization process.
- random_state =1 to ensure reproducibility.



```
# Logistic Regression with max_iter
log_reg = LogisticRegression(max_iter=30, random_state=1)
log_reg.fit(X_train, y_train)
y_pred_log_reg = log_reg.predict(X_test)
```

Logistic Regression- Metrics

Several key metrics were analysed to evaluate the performance of the Logistic Regression model. These metrics provide a comprehensive understanding of how well the model performs in predicting customer churn.

1. Classification Report:

Metric	Class 0 (Non-Churn)	Class 1 (Churn)
Precision	0.97	0.96
Recall	0.99	0.92
F1-Score	0.98	0.94
Support	1522	561

- Precision:
  - Class 0 (Non-Churn): 0.97 – The model is good at correctly identifying non-churns.
  - Class 1 (Churn): 0.96 – Precision is high, but marginally lower than for non-churn, indicating a few more incorrect churn predictions.
- Recall:
  - Class 0 (Non-Churn): 0.99 – The model excels at capturing nearly all non-churns
  - Class 1 (Churn): 0.92 – Lower recall for churned customers, means more actual churn cases were missed compared to non-churn.
- F1-Score:
  - Class 0 (Non-Churn): 0.98 – The model achieves a near-perfect balance between precision and recall for non-churn.
  - Class 1 (Churn): 0.94 – F1-score is slightly lower, reflecting the model struggling to balance precision and recall for churned customers.
- Support:
  - Class 0 (Non-Churn): 1522 – The larger number of non-churned customers helps the model perform more consistently for non-churn.
  - Class 1 (Churn): 561 – Fewer churned customers in the dataset may contribute to the slightly lower performance metrics for churns.

2. ROC AUC Score:

   ROC AUC score of 0.95 indicates that logistic regression has a strong ability to differentiate between churned and non-churned customers, making it good for predicting customer behaviour.

3. Confusion Matrix:

	Predicted: No Churn	**Predicted: Churn**
Actual: No	1531	21
Actual: Yes	46	515

- The model correctly identified 515 out of 561 actual churn cases.
- 46 churned customers were misclassified as non-churn.
- The model accurately predicted 1531 out of 1552 non-churn cases.
- 21 non-churned customers were misclassified as churn.

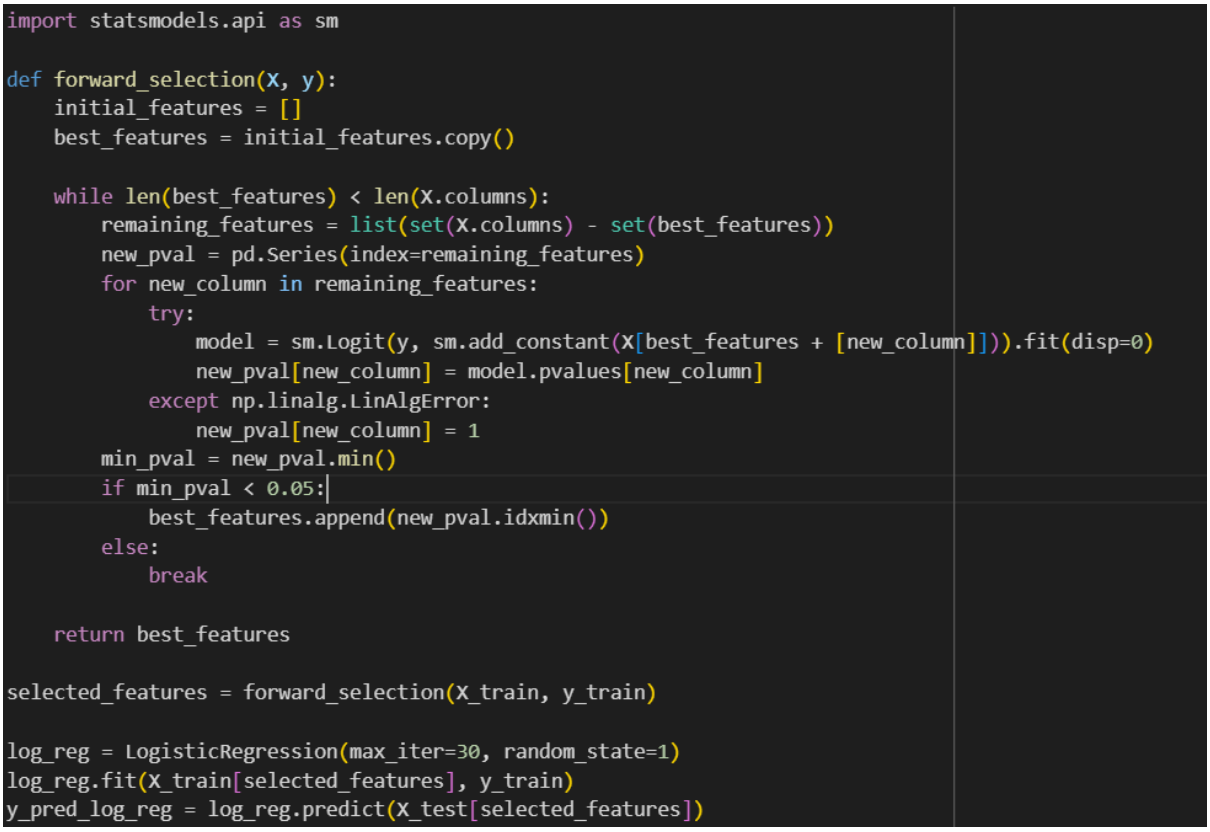
These results demonstrate the model's strong ability to accurately and reliably predict customer churn.

Logistic Regression- Forward

Forward regression starts off with no predictors and sequentially adds most significant features until no further improvements can be made.

Forward Regression - Model Training

Forward Regression model was implemented using a custom function that iteratively selects features based on their significance.



```
import statsmodels.api as sm

def forward_selection(X, y):
    initial_features = []
    best_features = initial_features.copy()

    while len(best_features) < len(X.columns):
        remaining_features = list(set(X.columns) - set(best_features))
        new_pval = pd.Series(index=remaining_features)
        for new_column in remaining_features:
            try:
                model = sm.Logit(y, sm.add_constant(X[best_features + [new_column]])).fit(disp=0)
                new_pval[new_column] = model.pvalues[new_column]
            except np.linalg.LinAlgError:
                new_pval[new_column] = 1
        min_pval = new_pval.min()
        if min_pval < 0.05:
            best_features.append(new_pval.idxmin())
        else:
            break

    return best_features

selected_features = forward_selection(X_train, y_train)

log_reg = LogisticRegression(max_iter=30, random_state=1)
log_reg.fit(X_train[selected_features], y_train)
y_pred_log_reg = log_reg.predict(X_test[selected_features])
```

Forward Regression - Metrics

Several key metrics were examined to assess the performance of the Forward Regression model. These metrics provide a complete picture of how well the model predicts customer churn.

Metric	Class 0 (Non-Churn)	Class 1 (Churn)
Precision	0.87	0.7
Recall	0.91	0.62
F1-Score	0.89	0.66
Support	1552	561

1. Classification Report:

- Precision:
  - Class 0 (Non-Churn): 0.87 – The model is reasonably accurate in identifying non-churned customers.
  - Class 1 (Churn): 0.70 – Indicates a moderate level of false positives in churn prediction.

- Recall:
  - Class 0 (Non-Churn): 0.91 – The model is effective at capturing non-churns.
  - Class 1 (Churn): 0.62 – A lower recall indicates that some actual churn cases were not identified.

- F1-Score:
  - Class 0 (Non-Churn): 0.89 – The model strikes a good balance between precision and recall for non-churn.
  - Class 1 (Churn): 0.66 – The lower F1-score reflects challenges in balancing precision and recall for churned customers.

- Support:
  - Class 0 (Non-Churn): 1522 – The larger number of non-churned customers helps the model perform more consistently for non-churn.

- Class 1 (Churn): 561 – Fewer churned customers in the dataset may contribute to the slightly lower performance metrics for churns.

2. ROC AUC Score:

   The Forward Regression model's ROC AUC score of 0.76 indicates that it has a moderate ability to distinguish between churned and non-churned customers, providing a reasonable foundation for customer churn prediction.

3. Confusion Matrix:

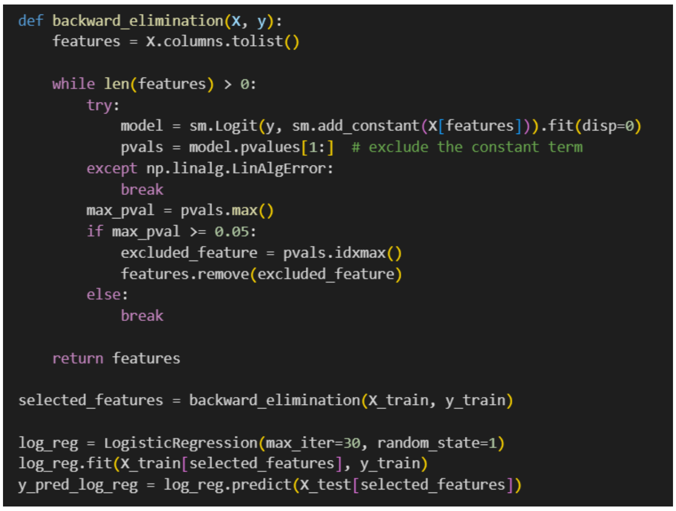
   Same as Logistic Regression.

Logistic Regression- Backward

Backward regression models begin with all available features and iteratively remove the least significant ones.

Backward Regression - Model Training

Backward Regression model was trained by removing features with the highest p-values sequentially until only statistically significant features remained.



```
def backward_elimination(X, y):
    features = X.columns.tolist()

    while len(features) > 0:
        try:
            model = sm.Logit(y, sm.add_constant(X[features])).fit(disp=0)
            pvals = model.pvalues[1:]  # exclude the constant term
        except np.linalg.LinAlgError:
            break
        max_pval = pvals.max()
        if max_pval >= 0.05:
            excluded_feature = pvals.idxmax()
            features.remove(excluded_feature)
        else:
            break

    return features

selected_features = backward_elimination(X_train, y_train)

log_reg = LogisticRegression(max_iter=30, random_state=1)
log_reg.fit(X_train[selected_features], y_train)
y_pred_log_reg = log_reg.predict(X_test[selected_features])
```

Backward Regression - Metrics

Several key metrics were examined to assess the performance of the Backward Regression model. These metrics provide a complete picture of how well the model predicts customer churn.

Metric	Class 0 (Non-Churn)	Class 1 (Churn)
Precision	0.97	0.96
Recall	0.99	0.92
F1-Score	0.98	0.94
Support	1552	561

ROC-AUC Score: 0.95

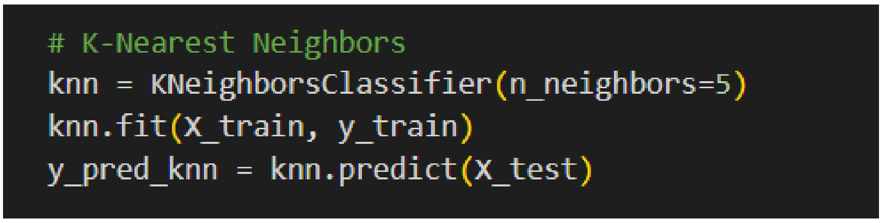
Backward Regression has identical numbers as Logistic Regression (Full).

K-Nearest Neighbours

K-Nearest Neighbours (KNN) is a model that classifies customers based on the majority class among their nearest neighbour. For Vortex, KNN helps in predicting customer churn by considering the similarities between customers based on their features.

K-Nearest Neighbours - Model Training

KNN model was trained using the 'KNeighborsClassifier' from the sklearn.neighbors library, with n_neighbors=5 to determine the number of neighbours considered during classification. The model was trained on the entire feature set, providing a straightforward approach to predicting churn.



```
# K-Nearest Neighbors
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train, y_train)
y_pred_knn = knn.predict(X_test)
```

K-Nearest Neighbours - Metrics

Metric	Class 0 (Non-Churn)	Class 1 (Churn)
Precision	0.82	0.56
Recall	0.87	0.48
F1-Score	0.84	0.52
Support	1552	561

1. Classification Report:

- Precision:
  - Class 0 (Non-Churn): 0.82 – The model is moderately accurate in identifying non-churned customers.
  - Class 1 (Churn): 0.70 – The lower precision reflects a higher rate of false positives for churn predictions.

- Recall:
  - Class 0 (Non-Churn): 0.87 – The model is effective at capturing non-churns.
  - Class 1 (Churn): 0.48 – A lower recall indicates that several actual churn cases were not identified.
- F1-Score:
  - Class 0 (Non-Churn): 0.84 – The model strikes a reasonable balance between precision and recall for non-churn.
  - Class 1 (Churn): 0.52 – The lower F1-score reflects challenges in balancing precision and recall for churned customers.

2. ROC AUC Score:

ROC AUC score of 0.67 indicates that the KNN model has limited ability to differentiate between churned and non-churned customers, suggesting room for improvement in predictive accuracy.

3. Confusion Matrix:

	Predicted: No Churn	**Predicted: Churn**
Actual: No	1344	208
Actual: Yes	292	269

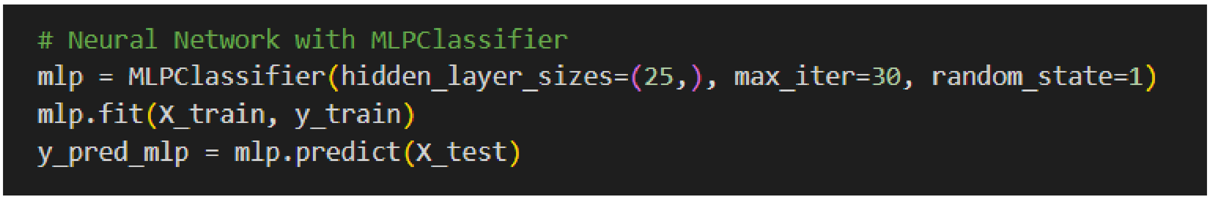
- KNN correctly identified 269 out of 561 actual churn cases.
- KNN misclassified 292 churned customers as non-churn.
- It also accurately predicted 1344 out of 1552 non-churn cases, misclassifying 208 non-churned customers as churn.

Neural Network

Neural Networks, specifically Multi-Layer Perceptrons (MLPs) is an advanced ML model capable of complex pattern recognition in data. For Vortex, the MLPClassifier was used to predict customer churn by learning from the intricate relationships between customer features.

Neural Network - Model Training

Training of the Neural Network was done using ‘MLPClassifier’ from the sklearn.neural_network library. ‘hidden_layer_size’ of 25 was found to be the best structure for this model. The model was trained with a ‘max_iter=30’.



```
# Neural Network with MLPClassifier
mlp = MLPClassifier(hidden_layer_sizes=(25,), max_iter=30, random_state=1)
mlp.fit(X_train, y_train)
y_pred_mlp = mlp.predict(X_test)
```

Neural Network - Metrics

Metric	Class 0 (Non-Churn)	Class 1 (Churn)
Precision	0.95	1.00
Recall	1.00	0.86
F1-Score	0.97	0.92
Support	1552	561

1. Classification Report:

- Precision:
  - <u>Class 0 (Non-Churn)</u>: 0.95 – The model is highly accurate in identifying non-churned customers.
  - <u>Class 1 (Churn)</u>: 1.00 – Perfect precision means that all predicted churn cases were correct.

- Recall:
  - Class 0 (Non-Churn): 1.00 – The model perfectly captures all non-churn cases.
  - Class 1 (Churn): 0.86 – A lower recall suggests that some actual churn cases were missed.
- F1-Score:
  - Class 0 (Non-Churn): 0.97 – The model strikes an ideal balance between precision and recall for non-churn.
  - Class 1 (Churn): 0.92 – The high F1-score reflects the model's strong performance in identifying churned customers.

2. ROC AUC Score:

ROC AUC score of 0.93 indicates that the Neural Network model is very good at distinguishing between churned and non-churned customers, making it a reliable tool for predicting customer churn.

3. Confusion Matrix:

	Predicted: No Churn	**Predicted: Churn**
Actual: No	1541	11
Actual: Yes	54	507

- NN correctly identified 507 out of 561 actual churn cases.
- NN misclassified 54 churned customers as non-churn.
- It also accurately predicted 1541 out of 1552 non-churn cases, misclassifying 11 non-churned customers as churn.

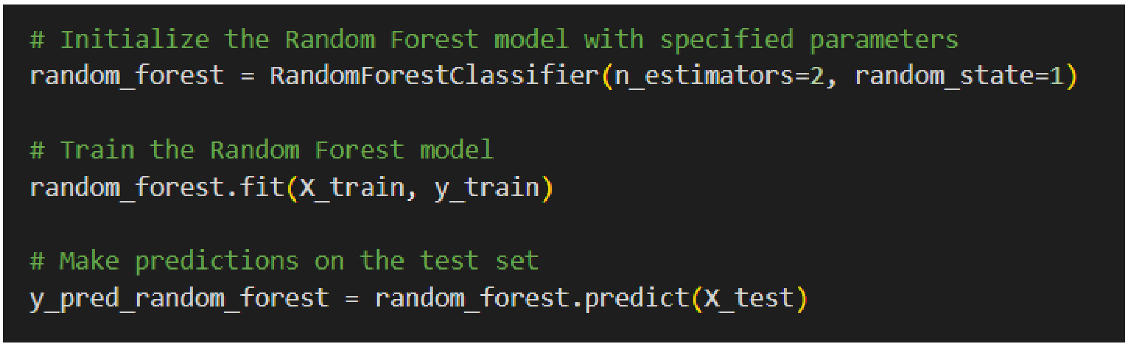
Random Forest

An ensemble learning model which builds multiple decision trees sampling data from the main dataset, which then carries out predictions on each decision tree and saves it. At the end of the modelling, it aggregates all decisions based on either majority voting of taking the mean of the predictions to provide with a final prediction metric which can be used to identify customer churns.

Random Forest - Model Training

The Random Forest model was implemented using the function 'RandomForestClassifier' from the sklearn.ensemble library. With 'n_estimators=2' to specify the number of trees in the random forest, the model has several advantages:

- Computation power needed to make predictions were far lesser.
- Overfitting was mitigated as there were fewer trees in the model. Since the cleaned dataset was relatively small, the model remained simple and generalized well.
- With the model achieving high precision, recall and ROC scores with 2 estimators suggests that the model has already captured primary patterns in the data. This also is a testament to effective feature engineering.
- High performance of the model with only 2 estimators suggests that the data worked on is highly separable and the decision boundaries are well-defined even with minimal aggregation.



```python
# Initialize the Random Forest model with specified parameters
random_forest = RandomForestClassifier(n_estimators=2, random_state=1)

# Train the Random Forest model
random_forest.fit(X_train, y_train)

# Make predictions on the test set
y_pred_random_forest = random_forest.predict(X_test)
```

Random Forest - Metrics

Metric	Class 0 (Non-Churn)	Class 1 (Churn)
Precision	0.95	1.00
Recall	1.00	0.94
F1-Score	0.99	0.97
Support	1552	561

4. Classification Report:

- Precision:
  - Class 0 (Non-Churn): 0.95 – The model is highly accurate in identifying non-churned customers.
  - Class 1 (Churn): 1.00 – Perfect precision means that all predicted churn cases were correct.

- Recall:
  - Class 0 (Non-Churn): 1.00 – The model perfectly captures all non-churn cases.
  - Class 1 (Churn): 0.94 – A high recall value shows that the model successfully identifies most churn cases.
- F1-Score:
  - Class 0 (Non-Churn): 0.99 – The model strikes an ideal balance between precision and recall for non-churn.
  - Class 1 (Churn): 0.97 – The high F1-score reflects the model's strong performance in both precision and recall for predicting churn.

5. ROC AUC Score:

With an ROC AUC score of 0.97, the Random Forest excels at distinguishing between churned and non-churned customers, making it a reliable predictor of customer churn.

6. Confusion Matrix:

	Predicted: No Churn	**Predicted: Churn**
Actual: No	1552	0
Actual: Yes	35	526

- Random Forest correctly identified 526 out of 561 actual churn cases.
- Random Forest misclassified 35 churned customers as non-churn.
- Random Forest also accurately predicted 1552 out of 1552 non-churn cases, not misclassifying any non-churn cases as churn cases.

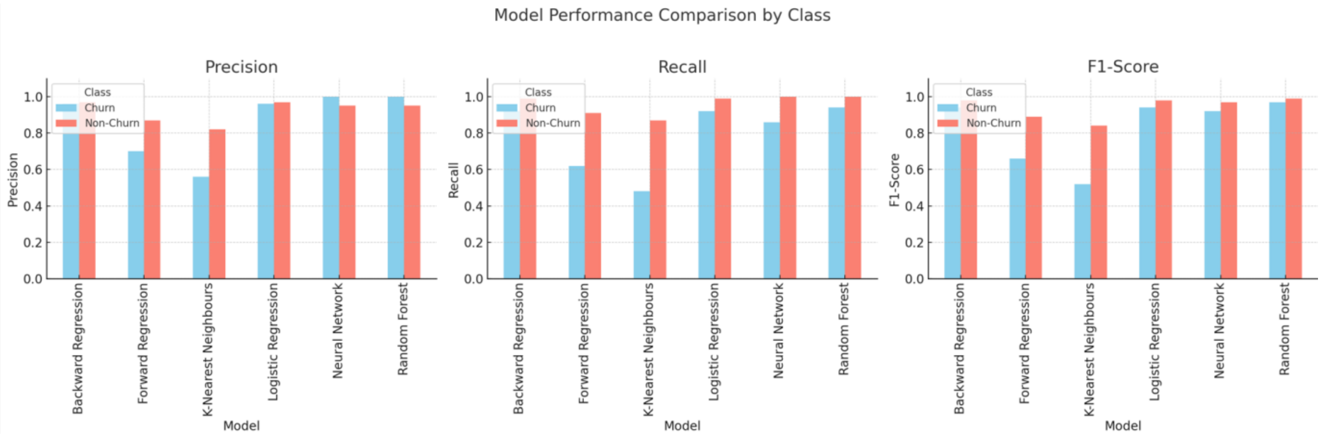
Model Comparison

To compare the effectiveness of the models used in this project and select one to predict customer churn, all metrics must be compared, including Precision, Recall, F1-Score, ROC AUC Score, and Confusion Matrix. These metrics provide a comprehensive understanding of how well each model identifies customers who are likely to churn versus those who are likely to remain.

1. Precision, Recall, and F1-Score:

The Precision, Recall, and F1-Score for each model across the two classes (Non-Churn and Churn) provide insights into the accuracy, coverage, and balance of the models' predictions. Below is a comparison table that highlights these metrics across all models.

Model	Class	Precision	Recall	F1-Score
Logistic Regression	Non-Churn	0.97	0.99	0.98
	Churn	0.96	0.92	0.94
Forward Regression	Non-Churn	0.87	0.91	0.89
	Churn	0.7	0.62	0.66
Backward Regression	Non-Churn	0.97	0.99	0.98
	Churn	0.96	0.92	0.94
K-Nearest Neighbours	Non-Churn	0.82	0.87	0.84
	Churn	0.56	0.48	0.52
Neural Network	Non-Churn	0.95	1	0.97
	Churn	1	0.86	0.92
Random Forest	Non-Churn	0.95	1	0.99
	Churn	1	0.94	0.97

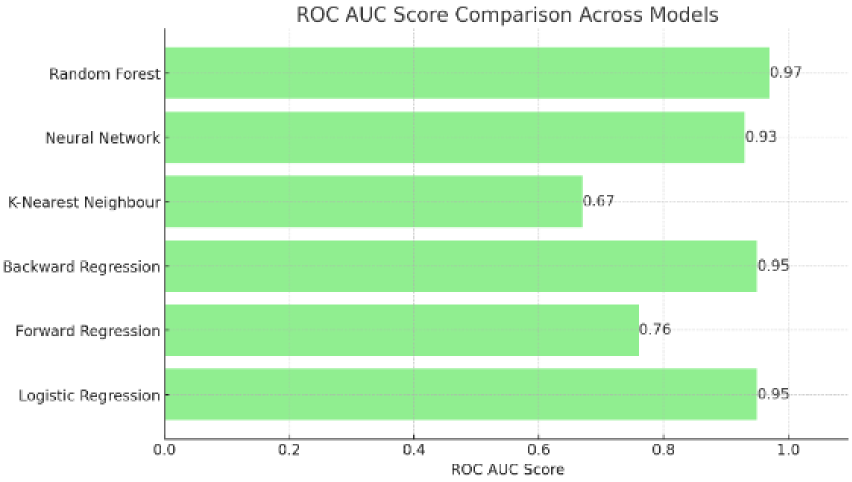


From the above table, we can notice that:

- Random Forest consistently performs the best, with the highest Precision, Recall and F1-Score across both classes. It achieves perfect Precision for the Churn class and has an impressive Recall of 0.94.
- Neural Network too performs well particularly in Precision for the Churn class, where it achieves a perfect score of 1.00. However it's Recall is slightly lower at 0.86.
- Logistic Regression and Backward Regression yield similar results, showcasing strong performance with high Precision and Recall scores.
- Forward Regression shows a moderate performance, especially with the Churn class where it's Recall and F1-Score are lower, indicating that it struggles more in identifying actual churn cases.
- K-Nearest Neighbours underperforms compared to the other models.

2. ROC AUC Score:

Model	ROC AUC Score
Logistic Regression	0.95
Forward Regression	0.76
Backward Regression	0.95
K-Nearest Neighbour	0.67
Neural Network	0.93
Random Forest	0.97



From the above table, we can notice that:

- Random Forest has the highest ROC AUC score of 0.97, indicating its ability to accurately predict customer churn.
- Logistic Regression and Backward Regression both achieve a strong ROC AUC score of 0.95, underscoring their reliability in churn prediction.
- Neural Network is a strong model for prediction tasks, coming in second with a ROC AUC score of 0.93.
- The moderate ROC AUC score of 0.76 for forward regression indicates its mediocre ability to distinguish between classes.
- With a ROC AUC of 0.67, K-Nearest Neighbours has the lowest score, indicating that it has more difficulty distinguishing between customers who have churned and those who have not.

3. Confusion Matrix:

Model	Predicted: No Churn	Predicted: Churn
Logistic Regression	Actual: No 1531	21
	Actual: Yes 46	515
Forward Regression	Actual: No 1406	146
	Actual: Yes 213	348
Backward Regression	Actual: No 1531	21
	Actual: Yes 46	515
K-Nearest Neighbours	Actual: No 1344	208
	Actual: Yes 292	269
Neural Network	Actual: No 1541	11
	Actual: Yes 54	507
Random Forest	Actual: No 1552	0
	Actual: Yes 35	526

From the above table, we can notice that:

- Random Forest performs better than other models with perfect predictions for non-churned customers (0 false positives) and very few false negatives (35 churned customers mistakenly classified as non-churn)
- With only 11 non-churned customers misclassified as churn and 54 churned customers misclassified as non-churn, neural networks perform exceptionally well as well.

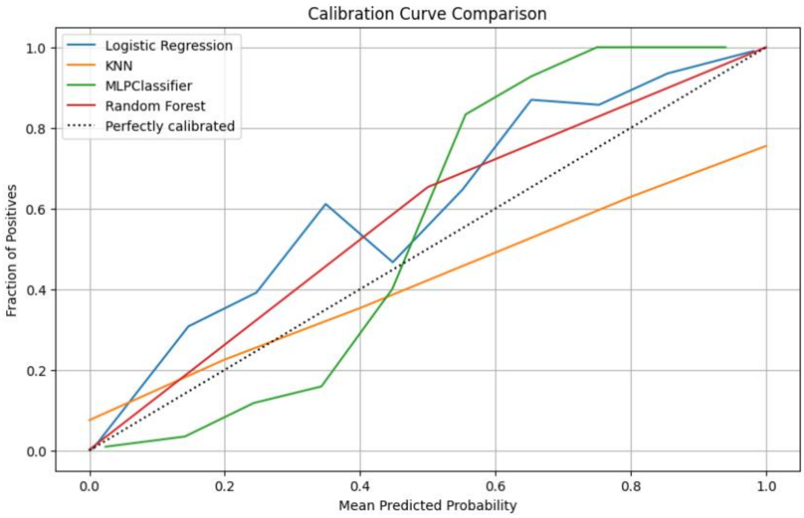
- The confusion matrices for Logistic Regression and Backward Regression are identical, showing a strong performance with 21 non-churned customers misclassified as churn and 46 churned customers misclassified as non-churn.
- Forward Regression and K-Nearest Neighbours perform worse and misclassify more customers.

In conclusion, Random Forest is by far the best model for predicting customer attrition across all metrics when it comes to accuracy, recall and precision. It is closely followed by the Neural Network, which performs admirably as well particularly in identifying non-churned customers.

Model Recommendation

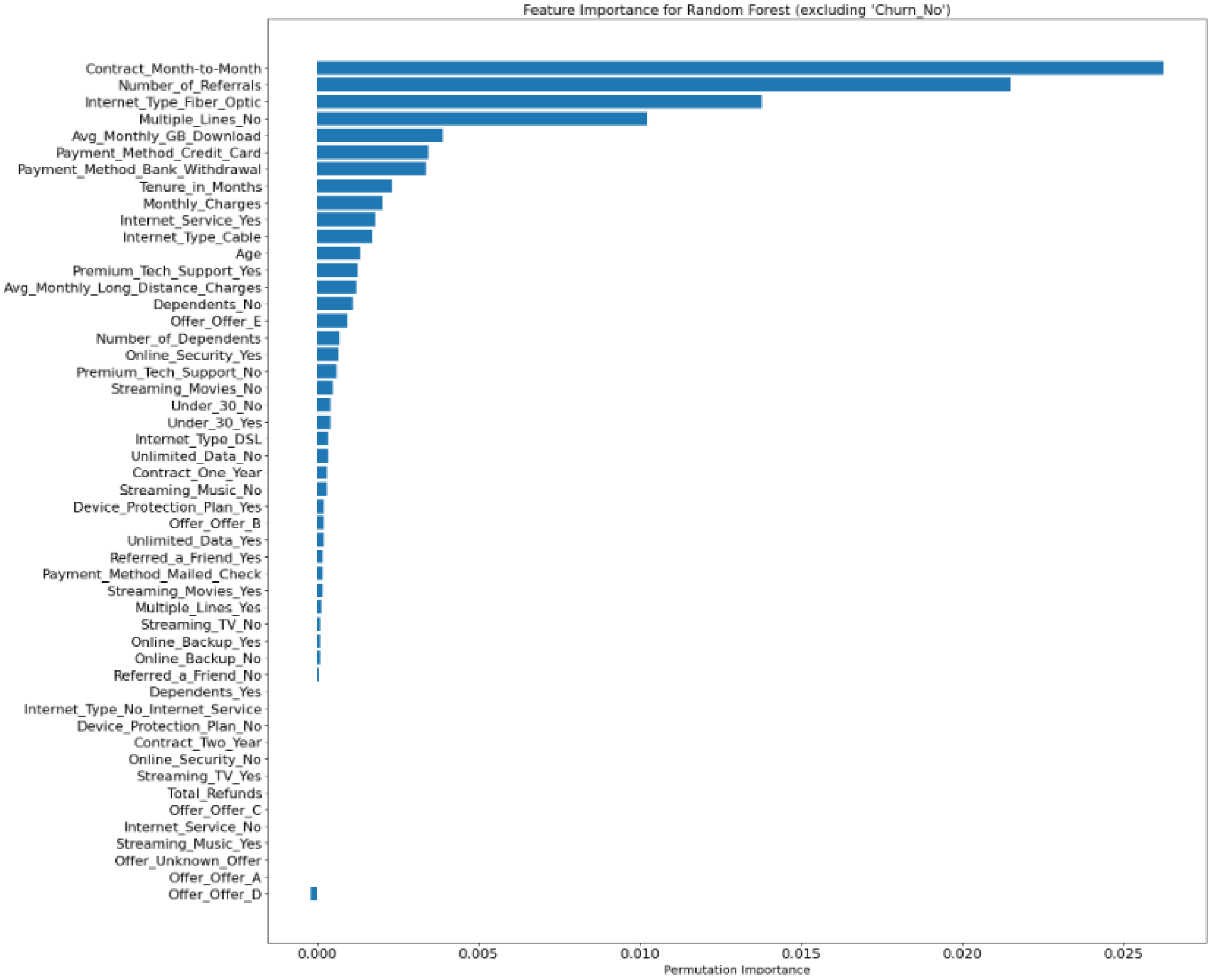
Random forest has established itself as the most dependable model that accurately predicts customer churn for the 7,000 customer database of Vortex. Being an ensemble learning technique, it builds multiple decision trees, each with unique subsets of data and predicts to deliver a final and highly accurate output.

- Robustness and Reliability: Random Forest is robust due to its ensemble nature. By relying on multiple decision trees, it reduces the risk of overfitting that might occur with a single decision tree. This leads to a model that generalizes well to unseen data, making it a reliable predictor in real-world scenarios.
- Comprehensive Pattern Recognition: The model excels at capturing complex patterns in the data. Its ability to consider multiple aspects of the data simultaneously allows it to identify nuanced relationships that might be missed by simpler models.
- Minimal Misclassifications: One of the standout features of Random Forest is its low rate of misclassifications. The model's design ensures that it accurately identifies the correct outcome for most customers, whether they are likely to churn or not. This precision minimizes errors, making Random Forest a trustworthy tool for businesses looking to reduce customer attrition.
- Calibration: Random Forest when implemented on the dataset calibrates extremely well with the data, which indicates that the predicted probabilities align well with actual outcomes.



Feature Importance

The feature importance that was derived from the Random Forest model which highlights the most critical factors which influences customer churn prediction. The top features that significantly impact the model's predictions are essential in understanding customer behaviour and implementing effective retention strategies. Let's discuss a few features below:



Key Features

- Contract Type: The "Contract: Month-to-Month" feature emerges as the most influential factor. According to Nyambura (2020), customers with a "month-to-month contract" are more likely to churn compared to those with longer-term contracts, indicating that the flexibility of monthly contracts may contribute to higher churn rates.
- Number of Referrals: The number of referrals is another top predictor of churn. A higher number of referrals seems to correlate with customer satisfaction and loyalty, reducing the likelihood of churn.
- Internet Type: The type of internet service, specifically "Internet_Type: Fiber_Optic," plays a crucial role. Fiber optic customers are less likely to churn,
- Multiple Lines: Customers with multiple lines also show a lower tendency to churn, likely because they have a greater investment in the service.

Suggestions & Recommended Next Steps

As Random Forest model has been identified as the most efficient predictor of customer churn for Vortex, our next steps would be to analyse the feature importance the model suggested and come up with strategic solutions to mitigate churn. Below are several key actions that can be implemented to leverage findings of the model and improve customer retention:

1. Targeted Retention Campaigns:

- Focus on Month-to-Month Customers: Considering the model indicates that customers on month-to-month contracts are at a higher risk of churn, targeted retention campaigns should be designed specifically for this group. Offering incentives to switch to longer-term contracts, such as discounts or added services, could reduce their likelihood of leaving.
- Encourage High Referring Customers: Targeting low-risk customers and encouraging them to bring in referrals is a great way to bring in more low-risk customers. We should also

incentivize the existing customers who bring in referrals, possibly through referral bonuses or exclusive offers.

2. Service Enhancement Initiatives:

- <u>Upgrade Internet Services</u>: Random Forest highlights that customers using fiber optic internet connections tend to churn less. Vortex must consider promoting fiber optic upgrades to customers on their internet services with possibility of promotional pricings, free installations, increase bandwidth and so on.
- <u>Optimize Multi-Line Plans</u>: customers who have multi-line plans tend to churn less, indicating that this segment of customers are loyal with Vortex. We could introduce more services along this line to bring more people in.

3. Personalized Communication:

- <u>Predictive Insights</u>: Considering Random Forest model is good with its predictive capabilities, we can identify high-risk customers. We can then personalize communication like tailored email campaigns, in-app notifications and other approaches to address individual concerns and improve CSAT.

4. Expand Predictive Modelling:

- <u>Additional Models</u>: While our Random Forest model has proven to be the most effective, it could also be beneficial to test other advanced models such as Gradient Boost (XGBoost), SMOTE- synthetically boosting and levelling of dataset to work with a larger dataset to see if any additional predictive powers could be unlocked.

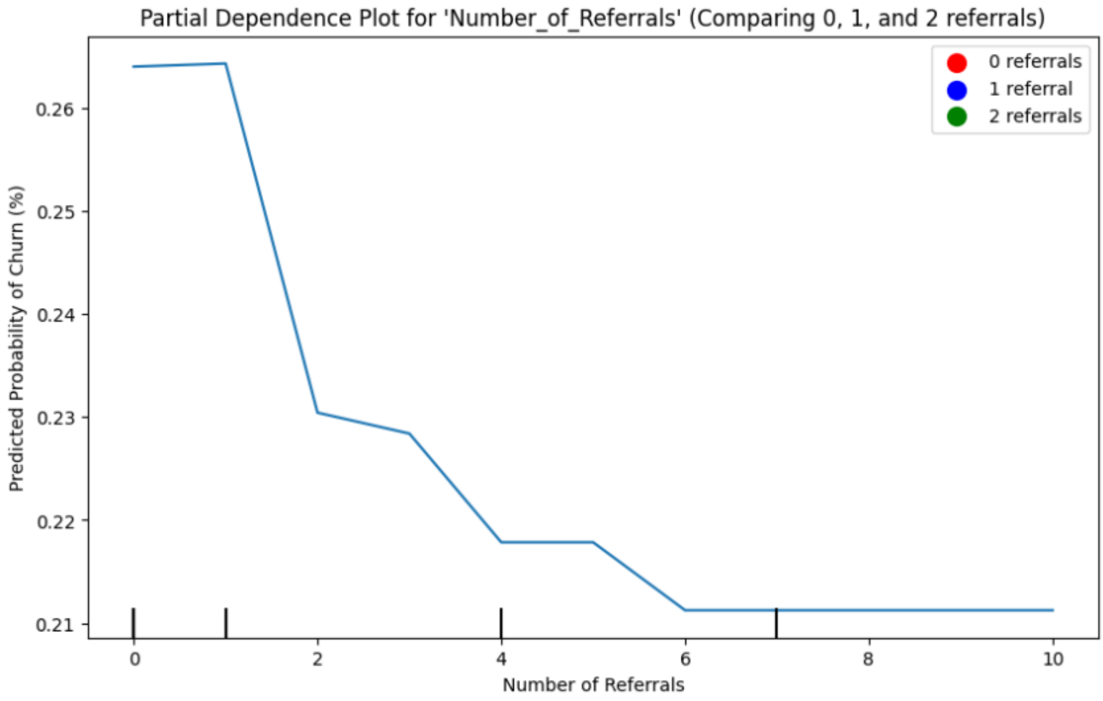
By implementing these solutions, Vortex can effectively reduce customer churn, enhance customer satisfaction, and ultimately drive long-term business growth.

Cause & potential Effect

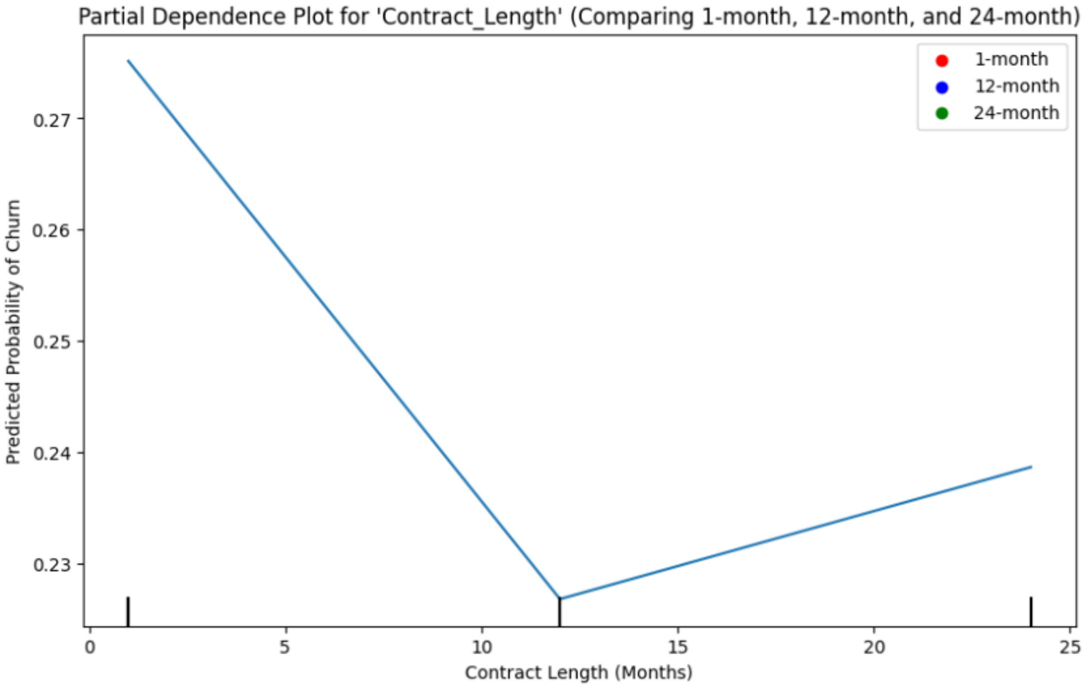
Let’s look at a few instances where an impact of a few features causes effects on customer churn percentages.

1. Impact of Referrals on Customer Churn: As seen below, he partial dependence plot for the "Number of Referrals" feature within the Random Forest model offers valuable insights into how customer referrals influence churn probability. The plot reveals a clear and significant inverse relationship: as the number of referrals increases, the predicted probability of churn decreases.

- Customers with zero referrals have the highest predicted probability of churn.
- When a customer makes just one referral, the churn probability drops significantly to approximately 23%.
- With two referrals, the predicted churn probability decreases further to around 22%.
- For customers who make four or more referrals, the churn probability stabilizes and reaches a low of 21%



2. Impact of Contract Length on Customer Churn: Below partial dependence plot for "Contract Length" provides key understanding into the impact of contract duration and the correlation with customer churn probability. The plot highlights the inverse relationship: as the contract length increases, the predicted probability of churn decreases significantly.
    - Customers with a 1-month contract length have the highest predicted probability of churn, around 27%.
    - Extending the contract length to 12 months results in a substantial reduction in churn probability, bringing it down to approximately 23%.
    - For customers on a 24-month contract, the churn probability slightly increases to around 24%, though it remains lower than the 1-month contract.



References

- FasterCapital. (n.d.). F1 Score. Retrieved from https://fastercapital.com/keyword/f1-score.html?cv=1
- GaborMelli. (n.d.). sklearn.neural_network Module. Retrieved from https://www.gabormelli.com/RKB/index.php?title=sklearn.neural_network_Module
- Nyambura, M. (2020). Predicting Customer Churn in Telecommunication: A Vodafone Case Study. Medium. Retrieved from https://medium.com/@nyamburam12/predicting-customer-churn-in-telecommunication-a-vodafone-case-study-introduction-04b65fa14056
- Patel, P., & Upadhyay, S. (2014). Classification algorithms in data mining [PDF]. Retrieved from https://core.ac.uk/download/pdf/84275226.pdf
- Pradeep, S., & Kumar, M. (2023). Optimized Transfer Learning-Based Dementia Prediction System for Rehabilitation Therapy Planning. International Journal for Research in Applied Science & Engineering Technology (IJRASET). Retrieved from https://www.ijraset.com/best-journal/optimized-transfer-learning-based-dementia-prediction-system-for-rehabilitation-therapy-planning
- Reichheld, F. F., & Sasser, W. E. (1990). Zero defections: Quality comes to services. Harvard Business Review, 68(5), 105-111.