Università degli studi di messina
MIFT Department

Machine Learning

Predicting Customer Churn for a Telecommunications Company

Supervising professor: Giacomo Fiumara

Student: Irmuunbilig Burenzevseg

Academic year 2023/2024

INTRODUCTION

This report addresses the critical challenge of customer churn in the telecommunications industry. Customer churn occurs when customers discontinue a company's services, resulting in lost revenue and potential business instability. Effectively predicting churn is essential for businesses, enabling them to implement proactive retention strategies, reduce losses, and preserve a loyal customer base.

In this project, we analyse a telecommunications company dataset comprising demographic information, service usage patterns, account details, and payment methods for 3,749 customers. The primary goal is to develop a robust machine learning model capable of accurately predicting which customers are at risk of churning, allowing the company to take preemptive actions to retain these customers.

1) Understanding the Dataset

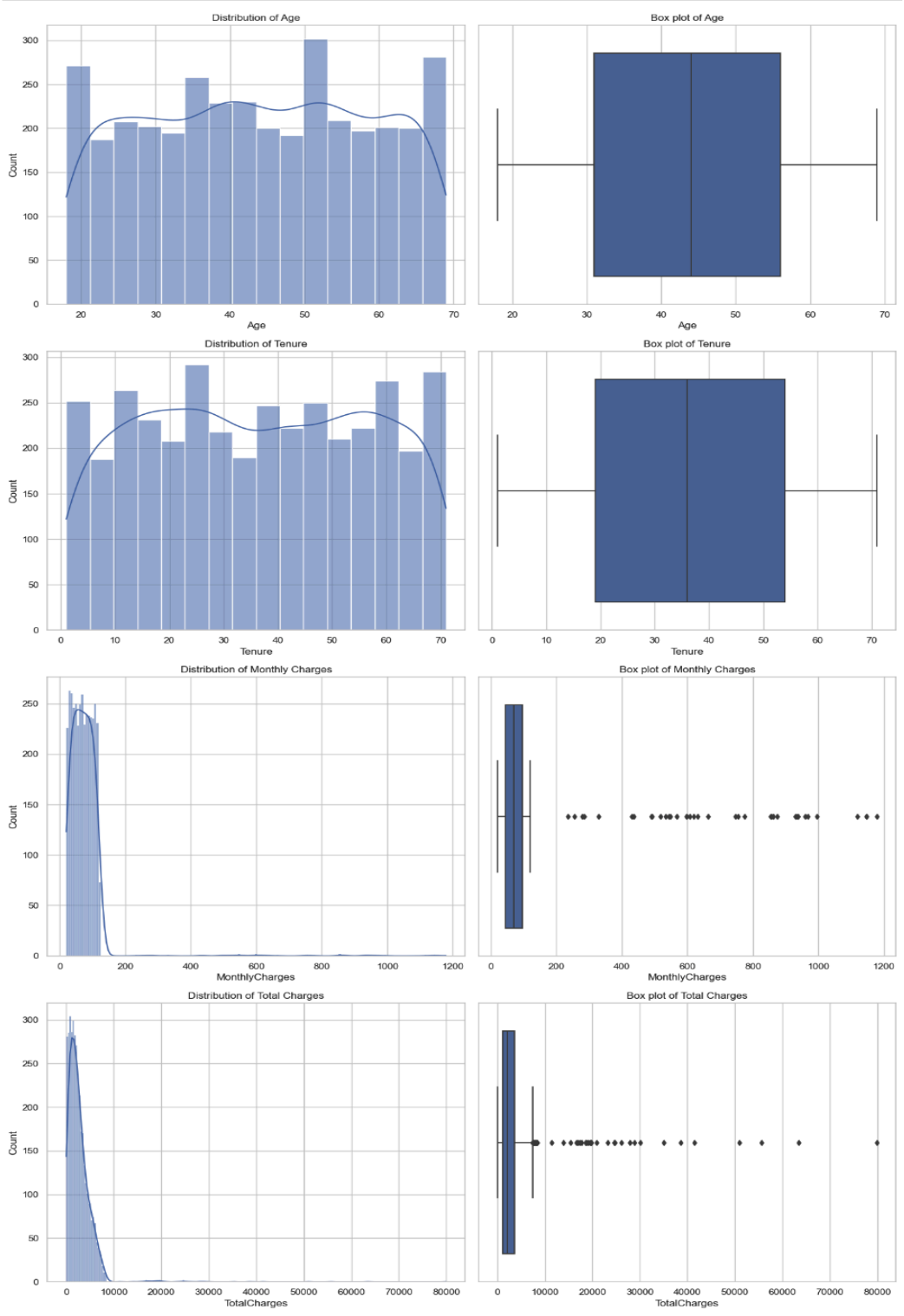
- **The dataset comprises several Features:**
    - **Numerical features:**
        - **Age**: This represents the customer's age. The values range from 18 to 69, with some missing values.
        - **Tenure**: Represents the number of months the customer has been with the company. The values range from 1 to 71 months.
        - **MonthlyCharges**: The monthly fee charged to the customer, ranges from $20 to $1,179.
        - **TotalCharges**: The total amount the customer has paid over their tenure. Values range from $13.19 to $79,951.
    - **Categorical features:**
        - **PaymentMethod**: Describes the payment method used by the

customer. Categories include Electronic check, Mailed check, Bank transfer, and Credit card. Some values are missing.

- **Gender**: Indicates whether the customer is male or female—possible values: Male, Female.
- **Service_Internet**: The type of internet service the customer subscribes to. Categories include DSL, Fiber optic.
- **Service_Phone**: Indicates whether the customer subscribes to phone service. Possible values: Yes, No.
- **Service_TV**: Indicates whether the customer subscribes to TV service. Possible values: Yes, No.
- **StreamingMovies**: Indicates whether the customer has access to streaming movies. Possible values: Yes, No.
- **StreamingMusic**: Indicates whether the customer has access to streaming music. Possible values: Yes, No.
- **OnlineSecurity**: Indicates whether the customer subscribes to online security services. Possible values: Yes, No.
- **TechSupport**: Indicates whether the customer subscribes to technical support services. Possible values: Yes, No.
- **CustomerID:** random string values mixed by the numbers and characters.
- **Contract:** Indicates customer's contract type with the company. Possible values: One Year, Two Year, Month-to-month

- Target:
  - **Churn**: The target variable indicates whether a customer has churned. Possible values: Yes, No.

Histograms and Box plots for numerical features:



Age:

- **Distribution**: The age range of customers is evenly spread between 18 and 69 years, with no noticeable bias toward a specific group. This suggests that age likely doesn't play a major role in whether customers churn.
- **Box Plot**: The age box plot shows no significant outliers, indicating that the data is well within a reasonable range for this feature.

Tenure:

- **Distribution**: The tenure feature, which represents the number of months a customer has been with the company, shows a roughly uniform distribution from 1 to 71 months.
- **Box Plot:** The box plot for tenure shows no significant outliers, confirming that most values fall within a normal range.

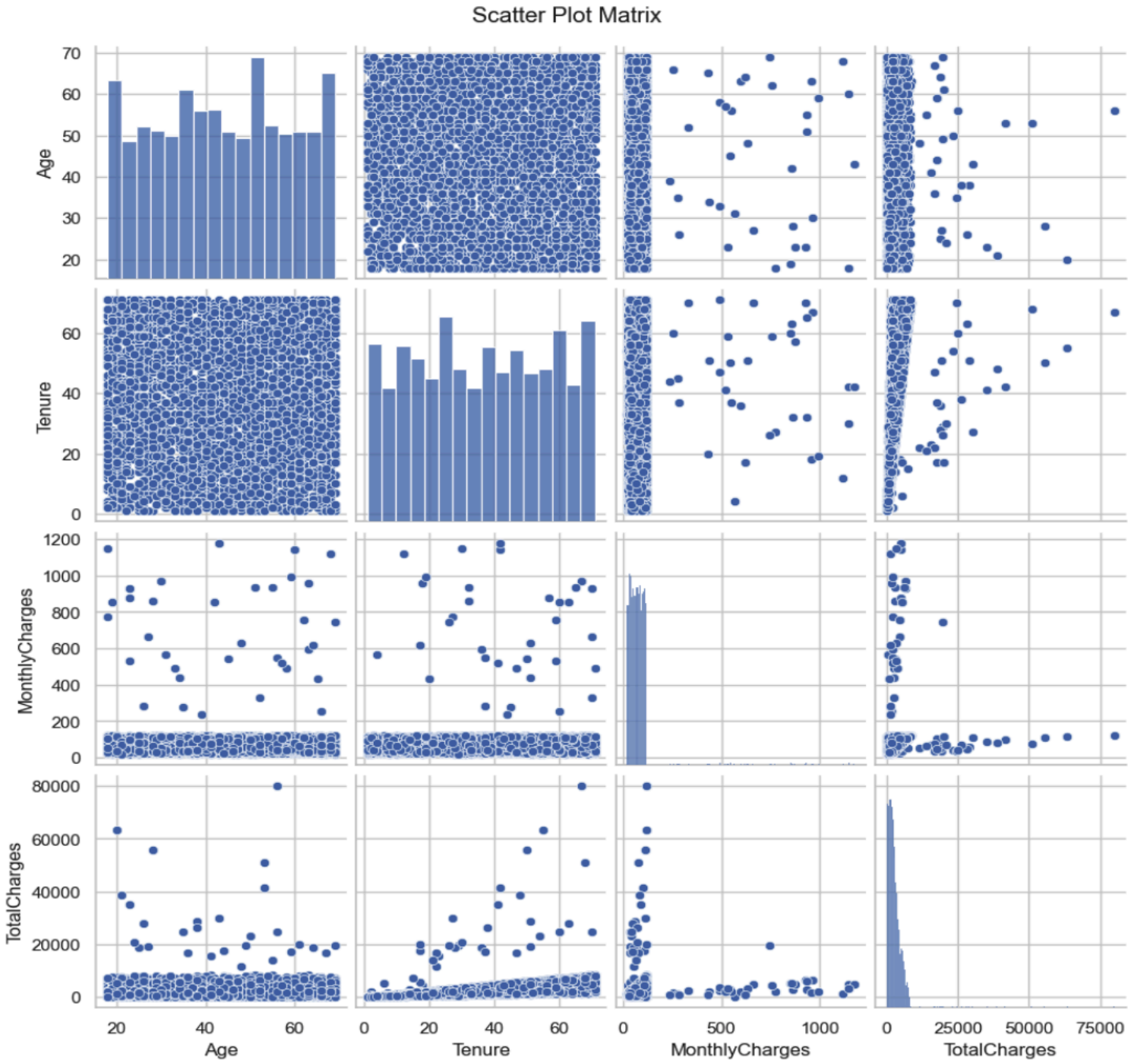
Monthly Charges:

- **Distribution**: Monthly charges are right-skewed, meaning most customers are charged between $20 and $200, with a few customers being charged significantly higher amounts.
- **Box Plot:** The box plot for monthly charges reveals several outliers, with a subset of customers being charged significantly higher amounts (up to $1,200). These high charges may relate to customers subscribing to premium services, which could influence their churn behaviour.

Total Charges:

- **Distribution:** The total charges exhibit a heavy right skew, with most customers accumulating lower charges while a few customers have very high total charges (up to $80,000).
- **Box Plot**: The box plot of total charges reveals significant outliers. These outliers likely represent long-term customers who have accumulated higher charges over time, which could correlate with their loyalty or engagement with multiple services.

Scatter plot matrix to visualize relationships between numerical features:



Observations:

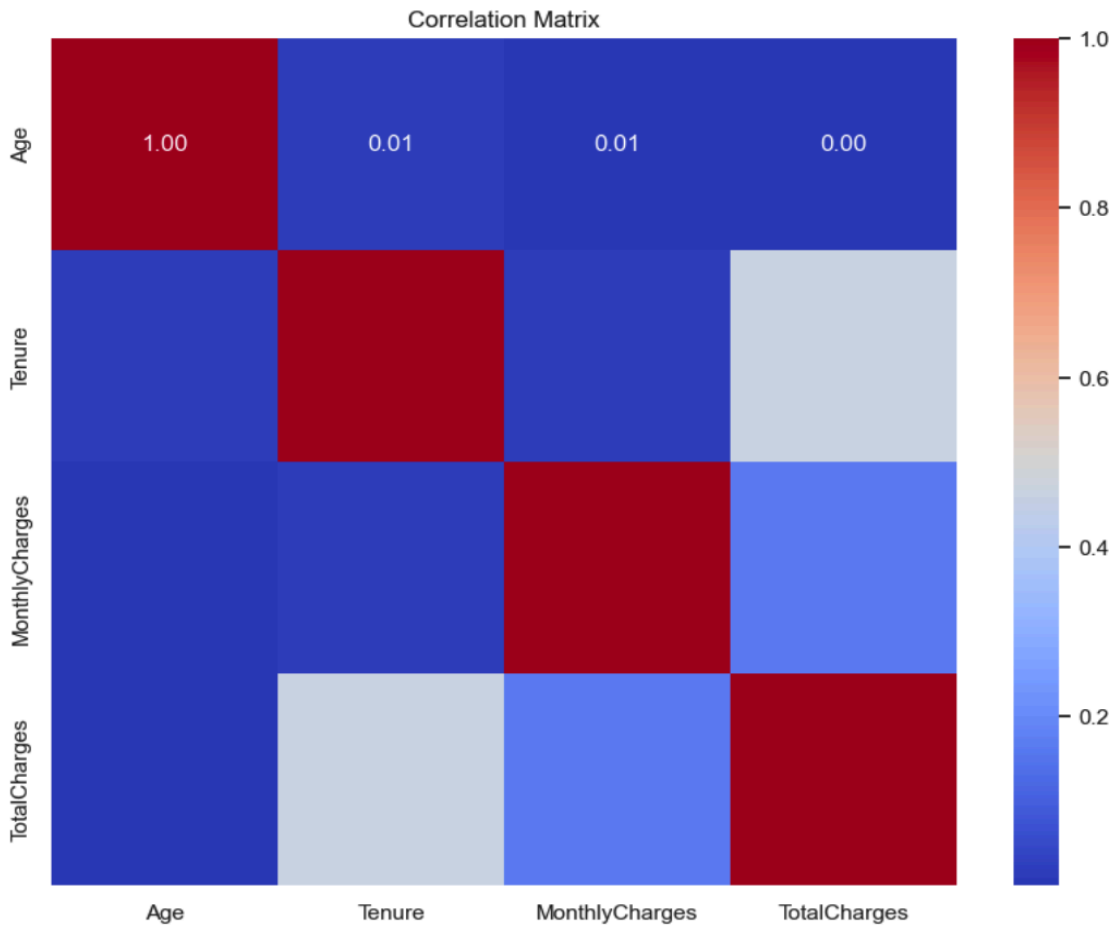
Age: The distribution of ages is relatively even, with no evident clusters or significant skewness. Scatter plots of age versus tenure, monthly charges, and total charges show no strong relationships. This indicates that a customer's age does not predict their tenure or charges, as data points are uniformly spread without clear trends.

Tenure: The histogram for tenure shows a roughly uniform distribution, indicating customers with varying lengths of tenure are well-represented. There is no strong correlation between tenure and monthly charges, as the scatter plot shows widely scattered points. However, tenure does have a clear relationship with total charges, which logically increase as tenure lengthens, reflecting accumulated charges over time.

MonthlyCharges: The distribution of monthly charges is skewed to the right, with the majority of customers paying lower amounts and fewer paying higher charges. While there is some correlation between monthly charges and total charges, the scatter plot shows variability, suggesting additional factors like tenure or customer behaviour influence total charges.

TotalCharges: The total charges distribution shows most customers have low total charges, with a few having significantly higher amounts. As expected, total charges generally increase with higher monthly charges, though there are outliers where customers have exceptionally high total charges, potentially indicating long-term customers with high monthly rates or anomalies.

Correlation matrix between numerical features:



General Insight:

- **Tenure** shows a moderate correlation with **TotalCharges**, indicating longer-tenure customers tend to accumulate higher charges.
- The correlation between **Tenure** and **Monthly Charges** is minimal, indicating that the time a customer has been with the company does not strongly impact their monthly charges.
- **Monthly Charges** have a positive correlation with **Total Charges** (approximately 0.3). This is logical because higher monthly charges contribute to higher total charges over time.

2) Data Preprocessing

Handling Missing Values:

```
# Checking for missing values and their proportions
missing_values = df.isnull().sum()
missing_percentage = (df.isnull().sum() / len(df)) * 100
missing_values_df = pd.DataFrame({'Missing Values': missing_values, 'Percentage': missing_percentage})

missing_values_df
```

	Missing Values	Percentage
CustomerID	0	0.000000
Age	187	4.987997
Gender	0	0.000000
Tenure	0	0.000000
Service_Internet	721	19.231795
Service_Phone	0	0.000000
Service_TV	0	0.000000
Contract	0	0.000000
PaymentMethod	187	4.987997
MonthlyCharges	0	0.000000
TotalCharges	0	0.000000
StreamingMovies	0	0.000000
StreamingMusic	0	0.000000
OnlineSecurity	0	0.000000
TechSupport	0	0.000000
Churn	0	0.000000

- In the dataset, a check for missing values across various features revealed the following:
    - **Age**: 187 missing values, constituting approximately 4.99% of the total entries.
    - **Service_Internet**: 721 missing values account for about 19.23% of the total dataset, indicating a significant portion of the data lacks internet service information.
    - **PaymentMethod**: 187 missing values, identical to the number of missing values for Age, also comprising about 4.99% of the total entries.

Imputing the Missing Values

```python
# Impute missing values in Age with the median
imputer_age = SimpleImputer(strategy='median')
df['Age'] = imputer_age.fit_transform(df[['Age']])

# Impute missing values in PaymentMethod with the most frequent value
imputer_payment = SimpleImputer(strategy='most_frequent')
df['PaymentMethod'] = imputer_payment.fit_transform(df[['PaymentMethod']]).ravel()

# Impute missing values in Service_Internet with the most frequent value
imputer_service_internet = SimpleImputer(strategy='most_frequent')
df['Service_Internet'] = imputer_service_internet.fit_transform(df[['Service_Internet']]).ravel()

# Verify missing values are handled
print(df.isnull().sum())
```

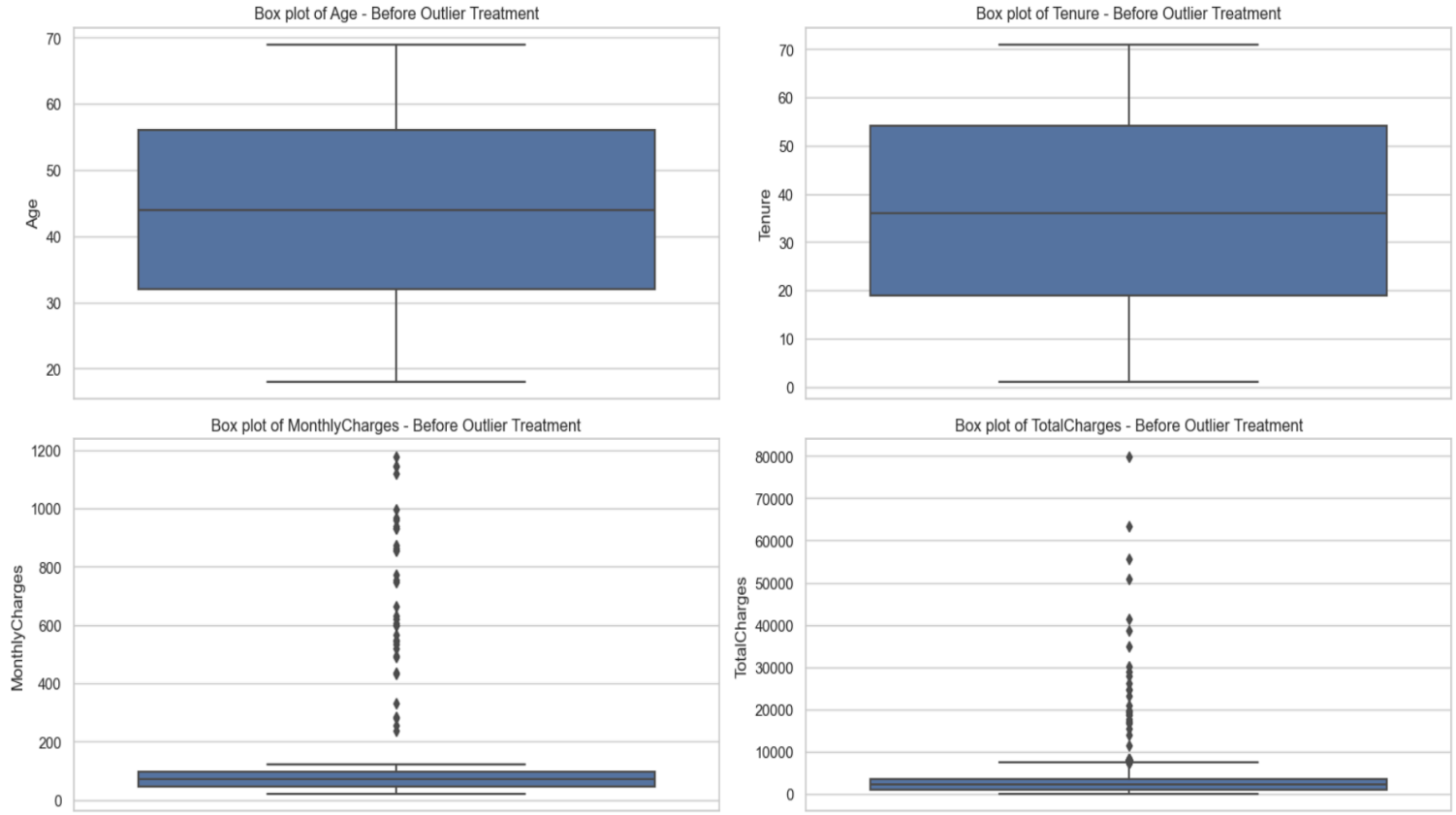
```
CustomerID          0
Age                 0
Gender              0
Tenure              0
Service_Internet    0
Service_Phone       0
Service_TV          0
Contract            0
PaymentMethod       0
MonthlyCharges      0
TotalCharges        0
StreamingMovies     0
StreamingMusic      0
OnlineSecurity      0
TechSupport         0
Churn               0
dtype: int64
```

- **Age**: Missing values in the `Age` column were imputed using the median of the existing age data. This approach helps to minimize the impact of extreme values and provides a central tendency that is robust to outliers.
- **PaymentMethod**: Missing values in the `PaymentMethod` column were replaced with the most frequent value (mode). This method is effective when a particular category dominates the feature, ensuring that the imputation aligns with the majority of existing data.
- **Service_Internet**: Similarly, missing values in the `Service_Internet` column were imputed with the most frequent value. This decision was made based on the assumption that the most common internet service status among the customers could reasonably represent the missing entries.

Outlier Treatment:

Outliers can distort statistical analyses and affect the performance of machine learning models, making it crucial to identify and treat them appropriately.

Box Plots Before Treatment:



The initial box plots for MonthlyCharges, and TotalCharges reveal the presence of significant outliers

- **MonthlyCharges:** The plot shows numerous data points far above the upper whisker, indicating that a small subset of customers are paying considerably higher monthly fees.
- **TotalCharges:** Similarly, TotalCharges exhibits a spread of outliers, suggesting that a few customers have accumulated charges much higher than the majority.

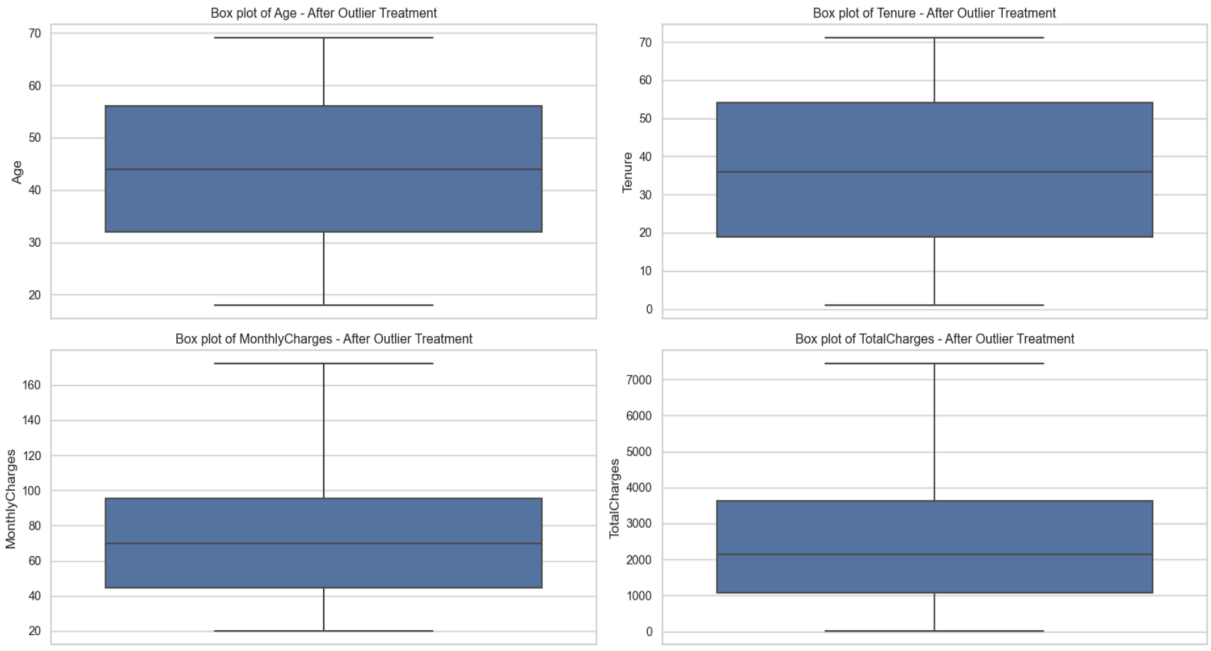
Outlier Treatment Methodology:

Interquartile Range (IQR) Method:

The IQR method was applied to the numerical features to identify and cap the outliers.

- **IQR Calculation:**
  - The lower bounds were computed as:
    - Q1 − 1.5 × IQR
  - upper bounds were computed as:
    - Q3 + 1.5 × IQR
  - Data points outside these bounds were replaced with the nearest boundary values, effectively capping the outliers.

Box plot After the Treatment:



- The extreme values in `MonthlyCharges` and `TotalCharges` have been mitigated. The maximum values for these features are now significantly lower, indicating successful outlier capping.
- **Age** and **Tenure** remain unchanged

Encoding Categorical Variables:

- In this part of the data preprocessing pipeline, the focus was on encoding categorical variables and preparing the dataset for model training.

```
from sklearn.preprocessing import OneHotEncoder

# One-hot encode categorical variables
categorical_features = ['Gender', 'Service_Internet', 'Service_Phone', 'Service_TV',
                        'Contract', 'PaymentMethod', 'StreamingMovies', 'StreamingMusic',
                        'OnlineSecurity', 'TechSupport', 'Churn']

df = pd.get_dummies(df, columns=categorical_features, drop_first=True)

# Convert boolean True/False to integer 1/0 for the encoded columns
boolean_columns = df.select_dtypes(include=['bool']).columns
df[boolean_columns] = df[boolean_columns].astype(int)


# Verify encoding
print(df.head())
```

```
                             CustomerID   Age  Tenure  MonthlyCharges  \
0  08729464-bde6-43bc-8f63-a357096feab1  56.0    13.0           71.88
1  af95bc95-baf4-4318-a21d-70d2ea3148b7  69.0    13.0          110.99
2  1fe7eee6-2227-4400-9998-4d993f4a60fd  46.0    60.0          116.74
3  f736fe7b-1b44-4acd-84c2-21c4aef648be  32.0    57.0           78.16
4  4b40d12d-7633-4309-96b8-aee675ea20ae  60.0    52.0           30.33

   TotalCharges  Gender_Male  Service_Internet_Fiber optic  Service_Phone_Yes  \
0        931.49            1                             0                  1
1       1448.46            1                             0                  0
2       6997.73            1                             1                  0
3       4452.13            0                             1                  1
4       1569.73            1                             1                  1

   Service_TV_Yes  Contract_One year  Contract_Two year  \
0               0                  1                  0
1               1                  0                  1
2               1                  0                  0
3               1                  0                  0
4               1                  0                  1

   PaymentMethod_Credit card  PaymentMethod_Electronic check  \
0                          0                               0
1                          0                               0
2                          0                               0
...
1                  0               0          0
2                  0               0          0
3                  0               1          0
4                  1               1          0
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

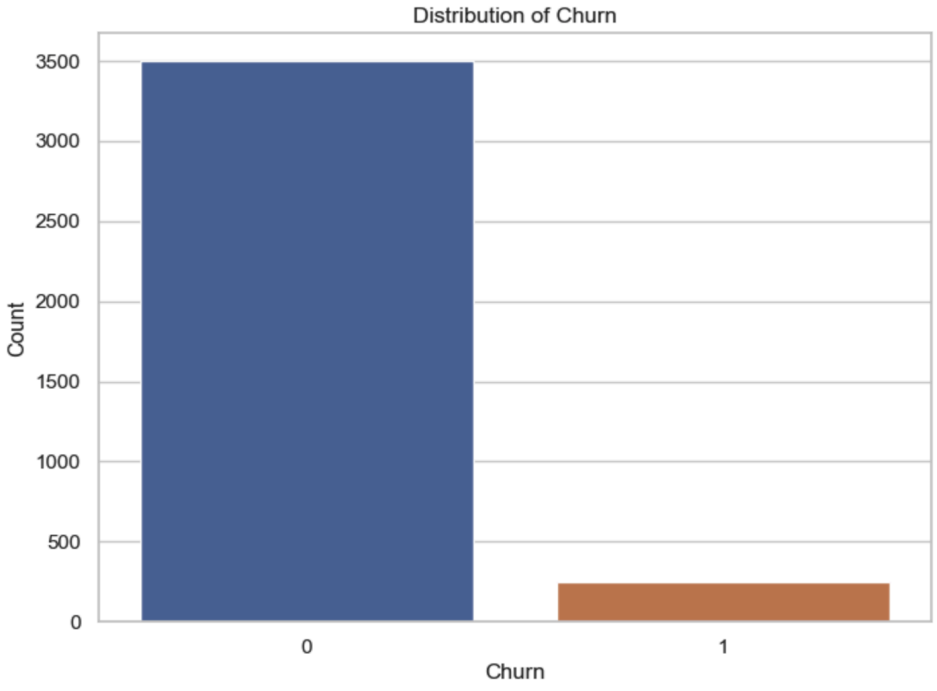
```
# Save the DataFrame to a CSV file
csv_file_path = '/Users/irmuunbilig/Desktop/cleaned_dataset.csv'
df.to_csv(csv_file_path, index=False)

csv_file_path
```

- **Categorical Features:** Gender, Service_Internet, Service_Phone, Service_TV, Contract, PaymentMethod, StreamingMovies, StreamingMusic, OnlineSecurity, TechSupport, and Churn.
- **One-Hot Encoding:** These categorical variables were transformed into numerical representations using **one-hot encoding**. This technique creates binary columns (0 or 1) for each category within a feature, allowing the model to interpret them without assuming any ordinal relationship between the categories.
    - The normalization of numerical features has been postponed until the model-building phase
    - The dataset resulting from the Data Preprocessing (Step 2) has been saved as a CSV file.

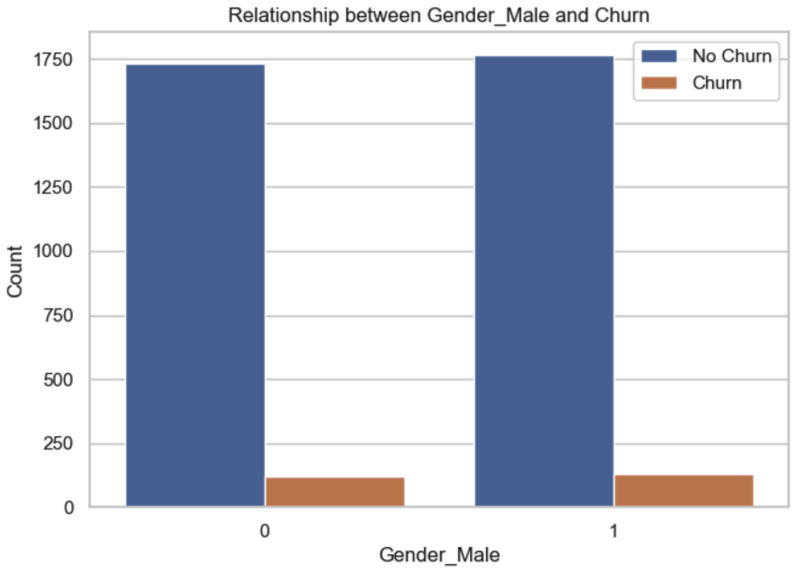
3) Exploratory Data Analysis (EDA)

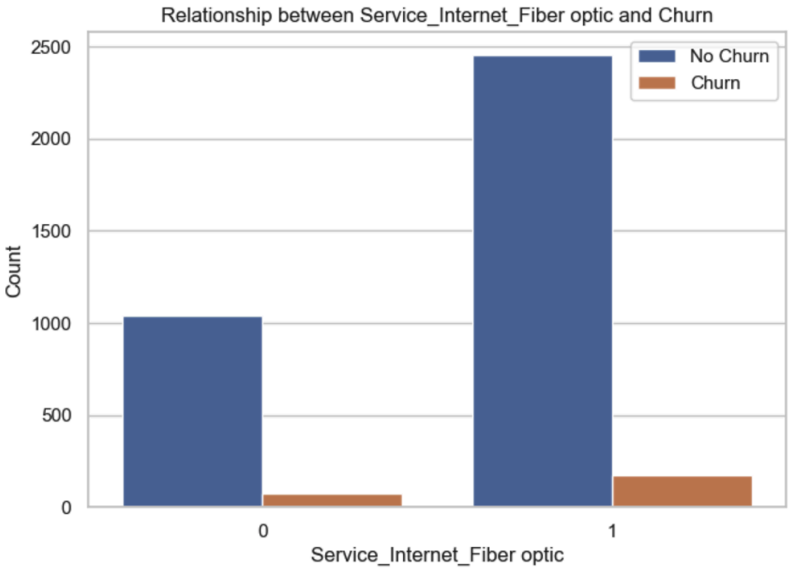
Churn Analysis:

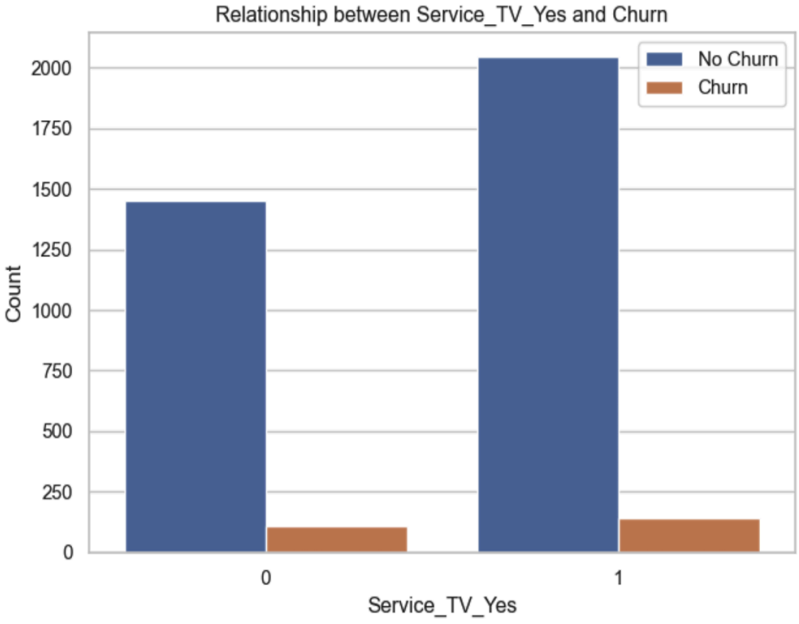
- Churn is heavily imbalanced with a significantly higher number of customers not churning.

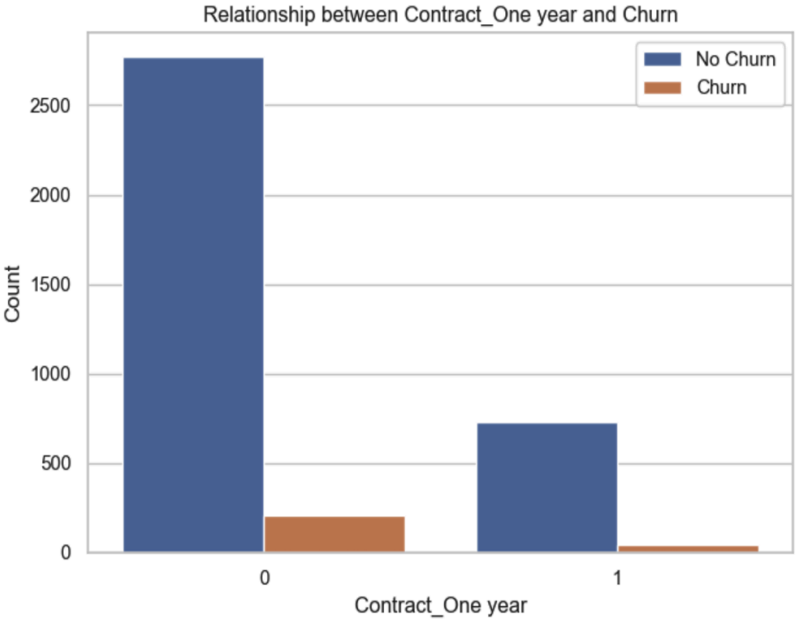


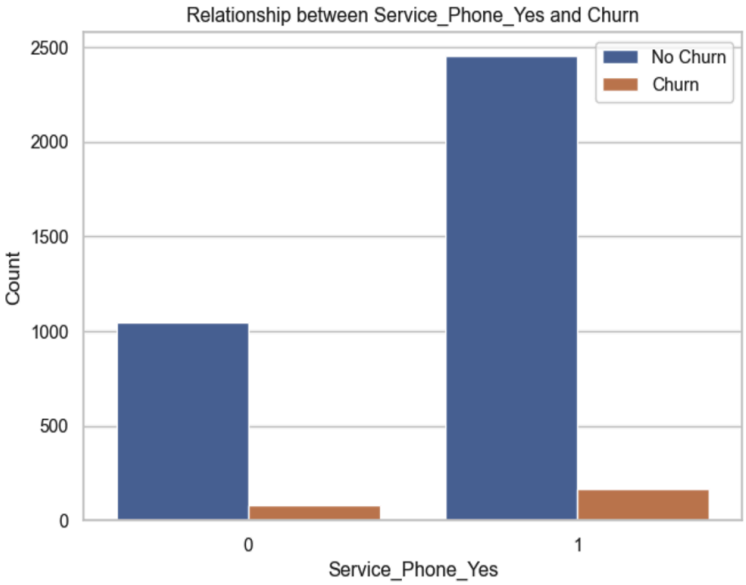
Categorical Feature Relationships with Churn:

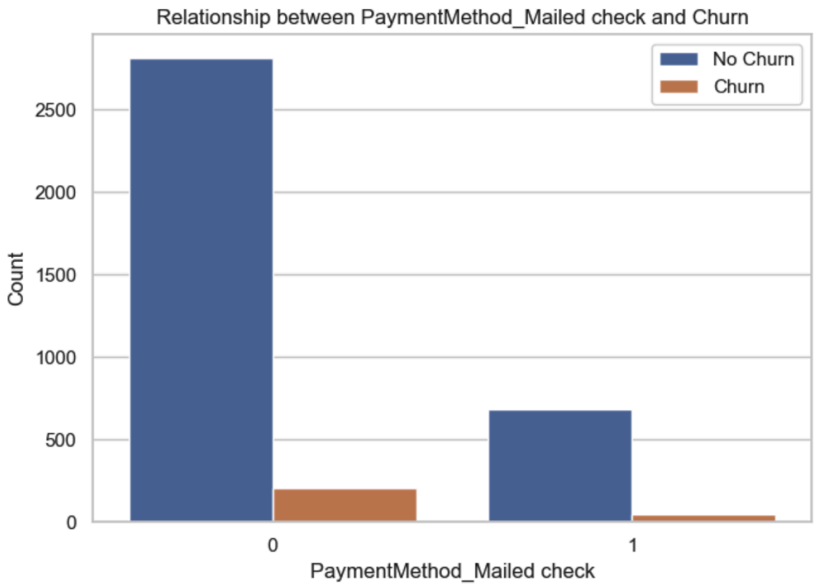


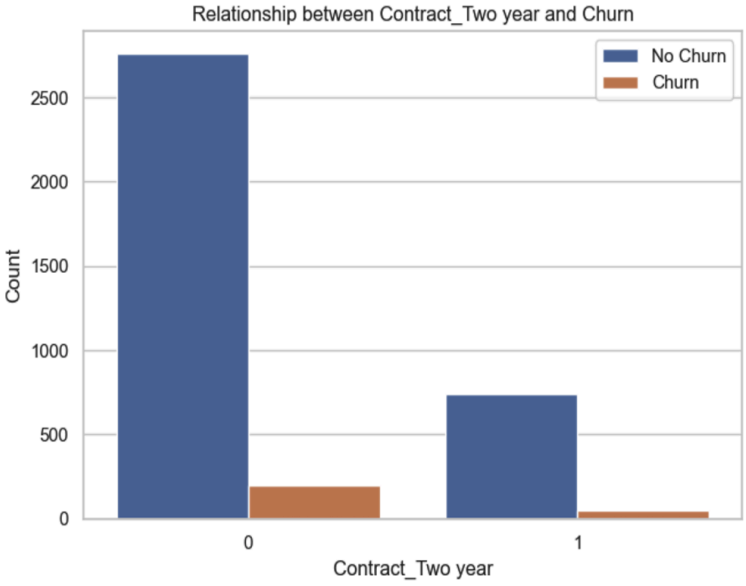


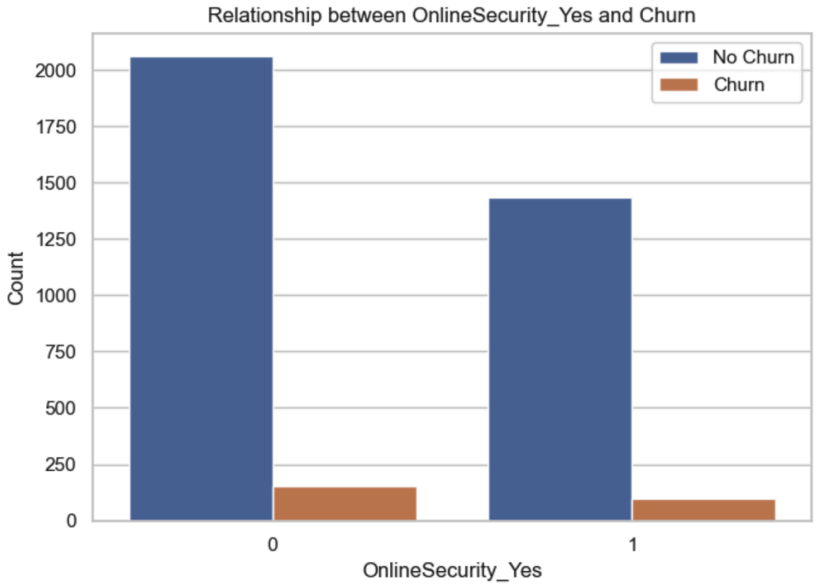


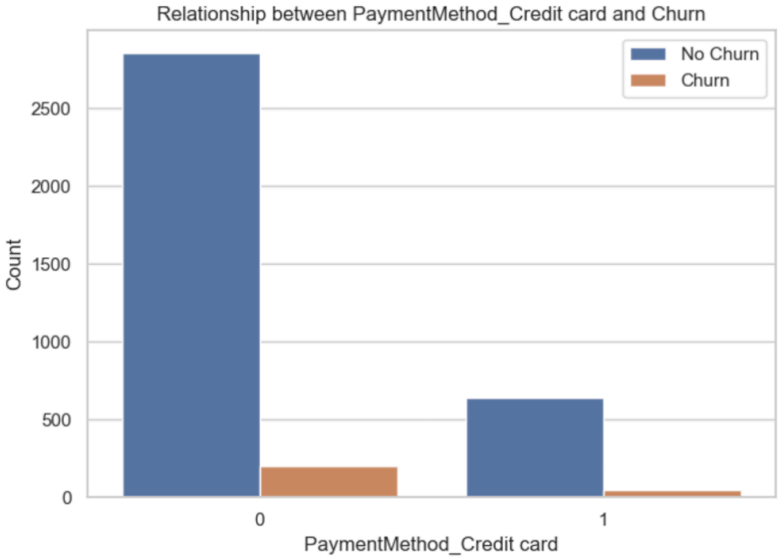


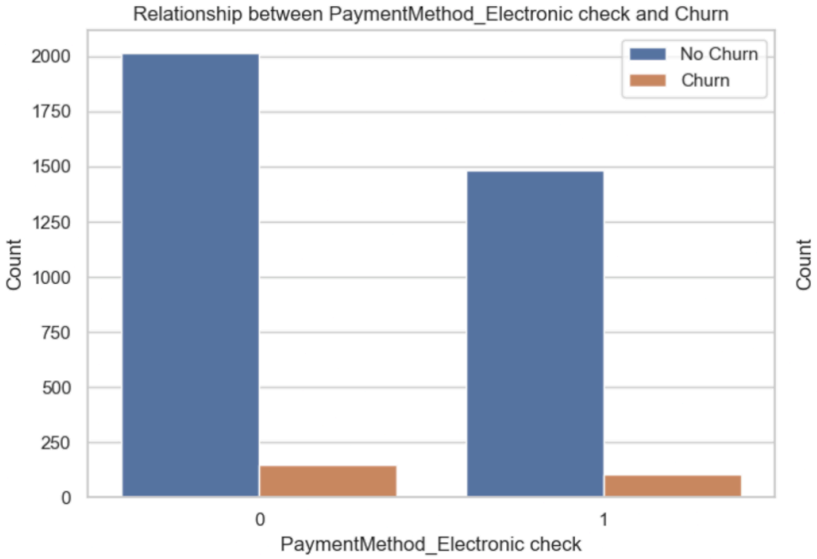

Relationship between Service_Phone_Yes and Churn
No Churn
Churn
Count
2500
2000
1500
1000
500
0
0
1
Service_Phone_Yes

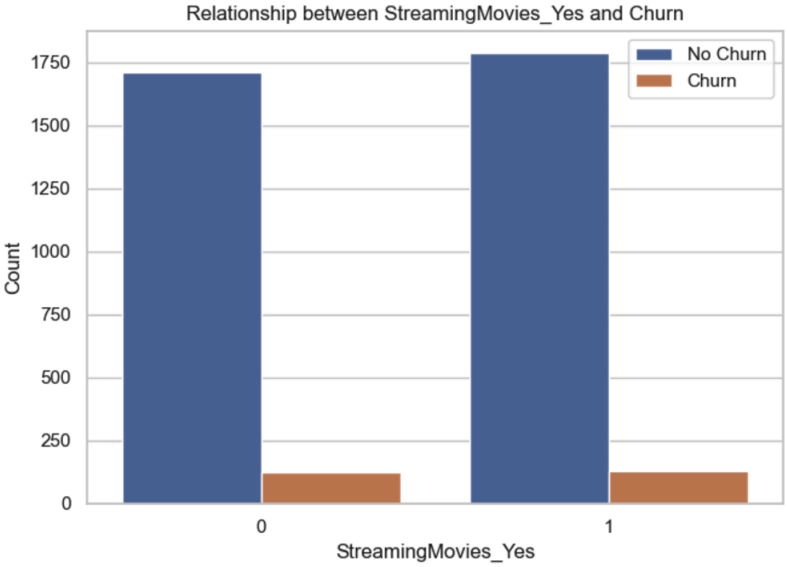

Relationship between PaymentMethod_Mailed check and Churn
No Churn
Churn
Count
2500
2000
1500
1000
500
0
0
1
PaymentMethod_Mailed check

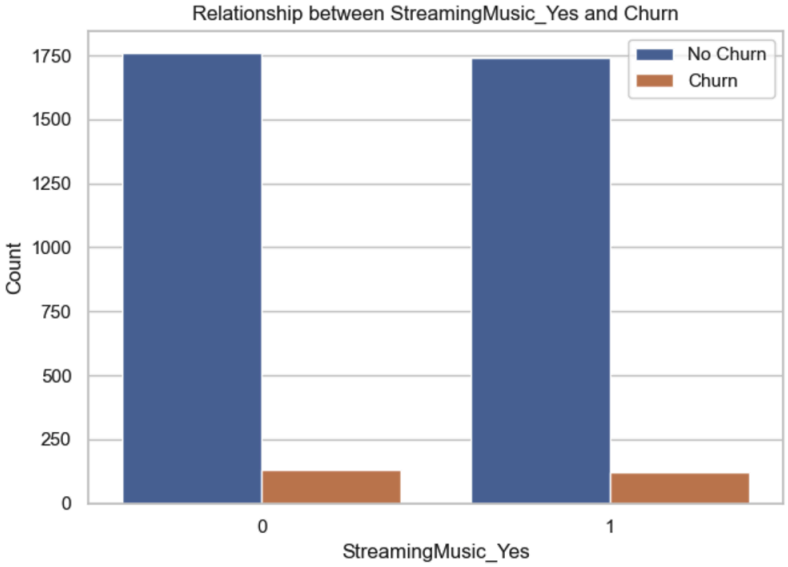

Relationship between Contract_Two year and Churn
No Churn
Churn
Count
2500
2000
1500
1000
500
0
0
1
Contract_Two year


Relationship between OnlineSecurity_Yes and Churn
No Churn
Churn
Count
2000
1750
1500
1250
1000
750
500
250
0
0
1
OnlineSecurity_Yes

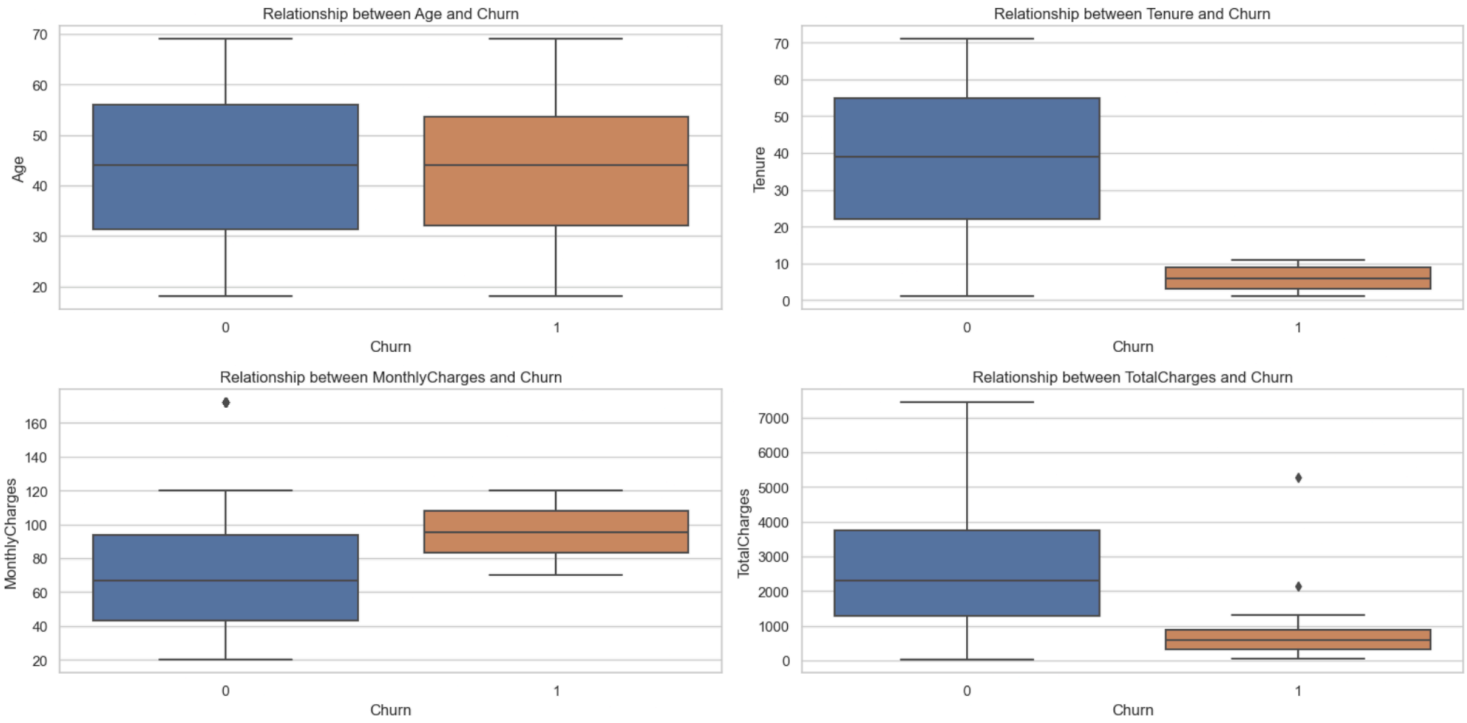

Relationship between PaymentMethod_Credit card and Churn
No Churn
Churn
Count
2500
2000
1500
1000
500
0
0
1
PaymentMethod_Credit card


Relationship between PaymentMethod_Electronic check and Churn
No Churn
Churn
Count
2000
1750
1500
1250
1000
750
500
250
0
0
1
PaymentMethod_Electronic check


Relationship between StreamingMovies_Yes and Churn
No Churn
Churn
Count
1750
1500
1250
1000
750
500
250
0
0
1
StreamingMovies_Yes


Relationship between StreamingMusic_Yes and Churn
No Churn
Churn
Count
1750
1500
1250
1000
750
500
250
0
0
1
StreamingMusic_Yes

Numerical Variables and Churn relationship:



Age and Churn:

- There is no significant difference in age between customers who churn and those who don't. The median ages for both churned and non-churned customers are very close, indicating that age might not be a strong predictor of churn.

Tenure and Churn:

- Tenure has a clear relationship with churn. Customers with shorter tenures (i.e., less time with the service) are much more likely to churn, while those who have been with the service for longer tend to stay. This suggests that retention efforts may need to focus on newer customers to prevent early churn.

Monthly Charges and Churn:

- Customers who churn tend to have slightly higher monthly charges compared to those who stay. This could indicate that customers paying higher monthly fees

are less satisfied and more likely to leave, potentially due to cost concerns.

Total Charges and Churn:

- Customers with lower total charges have a higher churn rate. This could be related to the tenure effect—those who churn earlier naturally accumulate fewer total charges. On the other hand, customers with high total charges are more likely to stay, potentially because they have a higher investment in the service.

Hypothesis Testing for Continuous and Categorical Features:

```
[35]: (                t_statistic        p_value
       Age               -0.210559   8.332430e-01
       Tenure           -26.158284  1.118112e-138
       MonthlyCharges    13.592804   4.106740e-41
       TotalCharges     -17.629702   6.826385e-67,
                                   chi2_statistic   p_value
       CustomerID                     3749.000000  0.492321
       Gender_Male                       0.117929  0.731292
       Service_Internet_Fiber optic      0.046761  0.828799
       Service_Phone_Yes                 1.292350  0.255615
       Service_TV_Yes                    0.069528  0.792026
       Contract_One year                 1.041092  0.307568
       Contract_Two year                 0.007072  0.932980
       PaymentMethod_Credit card         0.150893  0.697683
       PaymentMethod_Electronic check    0.000000  1.000000
       PaymentMethod_Mailed check        0.310064  0.577641
       StreamingMovies_Yes               0.042528  0.836617
       StreamingMusic_Yes                0.268162  0.604568
       OnlineSecurity_Yes                0.478169  0.489253
       TechSupport_Yes                   1.145177  0.284561)
```

1. T-Statistic and p-Value (for Continuous Variables)

- **Purpose**: Compares the means of two groups (churned and not churned customers) for numerical features.
- **Null Hypothesis (H_0)**: There's no significant difference in the mean of the feature between the two groups.
- **p-Value:**
  This value helps determine the significance of your results.
  - **p < 0.05:** The result is statistically significant, meaning you can reject the

null hypothesis.

- **p ≥ 0.05:** The result is not statistically significant, meaning you cannot reject the null hypothesis.
  - **Age**:
    - **t-statistic**: -0.210559
    - **p-value**: 0.833243
    - **Conclusion**: The p-value is much greater than 0.05, indicating that there is no statistically significant difference in age between churned and non-churned customers.
  - **Tenure**:
    - **t-statistic**: -26.158284
    - **p-value**: 1.118112e-138
    - **Conclusion**: There is a highly significant difference in tenure between churned and non-churned customers, with churned customers having much shorter tenures on average.
  - **MonthlyCharges**:
    - **t-statistic**: 13.592804
    - **p-value**: 4.106740e-41
    - **Conclusion**: Monthly charges are significantly higher for churned customers, with a very small p-value indicating strong evidence of a difference.
  - **TotalCharges**:
    - **t-statistic**: -17.629702
    - **p-value**: 6.826385e-67
    - **Conclusion**: Total charges are significantly lower for churned customers, likely due to shorter tenure, with a highly significant result.

2. Chi-Square Statistic and p-Value (for Categorical Variables)

- **Purpose**: Tests the association between categorical features and the target variable (churn).
- **Null Hypothesis (H_0)**: The feature is independent of churn.
- **p-Value:**
  It indicates the significance of the results:
  - **p < 0.05:** The result suggests a significant relationship between the feature and the target variable.
  - **p ≥ 0.05:** The result is not statistically significant, suggesting no relationship between the feature and the target variable.

Summary:

- **Statistically Significant Features**:
  - **Tenure**, **MonthlyCharges**, and **TotalCharges** are highly statistically significant in predicting customer churn. These continuous features should be the primary focus when building churn prediction models.
- **Non-Significant Features**:
  - Many categorical features, such as **gender**, **contract type**, **payment method**, and other service-related features, do not show statistical significance in predicting churn.

These results suggest that continuous features like **tenure**, **monthly charges**, and **total charges** play a crucial role in predicting customer churn, while most categorical variables may have less impact on their own.

4) Feature Engineering

New Features Created:

The following features were created to enhance the predictive power of the customer churn model by capturing more complex relationships between variables and customer behaviours.

- **TotalServiceUsage:**
  - Combines multiple service indicators: Service_Phone_Yes,Service_Service_TV_Yes,StreamingMovies_Yes,StreamingMusic_Yes, OnlineSecurity_Yes, TechSupport_Yes and Service_Internet_Fiber optic. It sums the number of services each customer subscribes to, providing a measure of overall engagement with the company's services.
    - Customers with higher engagement across multiple services are often more integrated into the ecosystem, which may reduce their likelihood of churn. This feature helps capture the overall service utilization level of each customer.
- **MonthlyChargesPerService:**
  - This feature is calculated by dividing the customer's total monthly charges by the total number of services they use.
    - Highlights customers who pay more per service. Higher costs may lead to higher churn if perceived value is low.
- **TenurePerService**:
  - This feature is derived by dividing the customer's tenure by the total number of services they use.
    - The purpose is to provide insights into customer loyalty by showing the average tenure per service.
- **Loyalty_Index**:
  - The product of TotalServiceUsage and tenure.
    - Combines service engagement and loyalty. Higher values suggest customers who are both long-term and highly engaged, reducing churn likelihood.

Impact:
These newly engineered features directly impact both model accuracy and interpretability. By capturing interactions and aggregating customer behaviours, the model can better predict customer churn and provide more actionable insights for decision-makers.

- Feature Engineering (Step 4) output has been saved as a CSV file.

5) Model Building

In this section, multiple machine-learning models were built and evaluated to predict customer churn. Several classifiers are tested, and their performance is evaluated using key metrics: accuracy, precision, recall, and F1-score. The goal is to identify the best-performing model for our customer churn dataset.

Step 1: Data Preparation

1. **Dropping Unnecessary Columns:**
   - CustomerID was dropped as it is irrelevant for predictive modelling.
2. **Defining Features and Target**:
   - `X`: All columns except `Churn_Yes` (target variable).
   - `y`: Target variable `Churn_Yes`, where `1` indicates churn and `0` indicates retention.
3. **Feature Scaling:**
   - Standardised all feature values using StandardScaler to ensure that features contribute equally to model training:
     - Mean of scaled features: 0
     - The standard deviation of scaled features: 1
4. **Train-Test Split:**
   - Data was split into training and testing sets:
     - **Training Set:** 75% of the data (2,811 samples)
     - **Testing Set:** 25% of the data (938 samples)
   - Ensures the model is tested on unseen data for performance evaluation.

Step 2: Model Selection

Three models were evaluated:

1. **Support Vector Machine (SVM):**
    - Suitable for high-dimensional datasets.
    - It uses hyperplanes to separate classes.
2. **Logistic Regression:**
    - Provides a baseline model for binary classification tasks.
    - Outputs probabilities, which can be used to adjust classification thresholds.
3. **K-Nearest Neighbors (KNN):**
    - The non-parametric model classifies samples based on the majority class of the nearest neighbours.

For each model:

1. **Training**:
    - The model was trained on the `X_train` and `y_train` datasets.
2. **Prediction**:
    - Predictions (`y_pred`) were made on `X_test`.

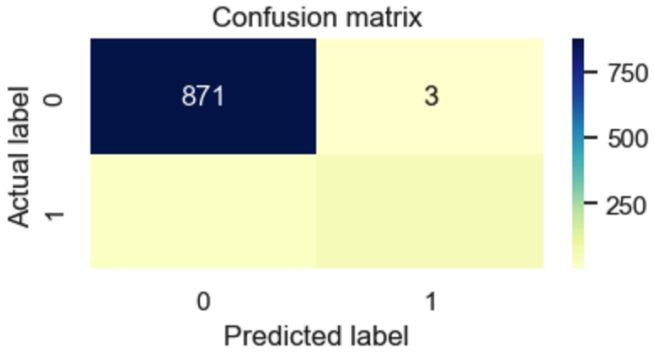
The evaluation metrics used are:

- **Accuracy:** The percentage of correct predictions out of all predictions.
- **Precision:** The ratio of correctly predicted positive observations to the total predicted positives.
- **Recall:** The ratio of correctly predicted positive observations to all observations in the actual class.
- **F1-Score:** a harmonic mean of precision and recall, providing a balanced measure of a model's accuracy in binary classification tasks, especially when dealing with imbalanced datasets.

Step 3: Results

1. Support Vector Machine (SVM):

```
Model: SVM
Accuracy: 97.23%
Precision: 93.18%
Recall: 64.06%
F1 Score: 75.93%

Classification Report:
              precision    recall  f1-score   support

           0       0.97      1.00      0.99       874
           1       0.93      0.64      0.76        64

    accuracy                           0.97       938
   macro avg       0.95      0.82      0.87       938
weighted avg       0.97      0.97      0.97       938
```



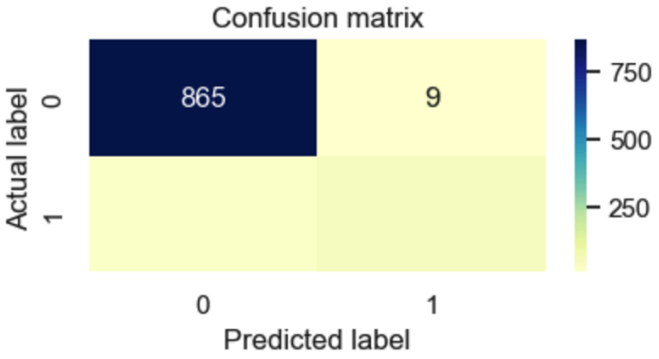
- **Accuracy:** 97.23%
- **Precision (Churn):** 93.18%
- **Recall (Churn):** 64.06%
- **F1-Score (Churn):** 75.93%
- **Confusion Matrix:**
  - **True Positives (TP):** 41 (correctly identified churners).
  - **False Negatives (FN):** 23 (missed churners).
  - **True Negatives (TN):** 871 (correctly identified non-churners).
  - **False Positives (FP):** 3 (non-churners identified as churners).
- **Observation:**
  - SVM has the highest precision for churners (93.18%), which minimizes false positives.

- The recall (64.06%) indicates some churners were missed.
- This model balances high accuracy and low false-positive rates, making it robust for churn prediction where minimizing false alerts is important.

---

2. Logistic Regression:

```
Model: Logistic Regression
Accuracy: 97.01%
Precision: 83.33%
Recall: 70.31%
F1 Score: 76.27%

Classification Report:
              precision    recall  f1-score   support

           0       0.98      0.99      0.98       874
           1       0.83      0.70      0.76        64

    accuracy                           0.97       938
   macro avg       0.91      0.85      0.87       938
weighted avg       0.97      0.97      0.97       938
```



- **Accuracy:** 97.01%
- **Precision (Churn):** 83.33%
- **Recall (Churn):** 70.31%
- **F1-Score (Churn):** 76.27%
- **Confusion Matrix:**
  - **True Positives (TP):** 45.
  - **False Negatives (FN):** 19.
  - **True Negatives (TN):** 865.

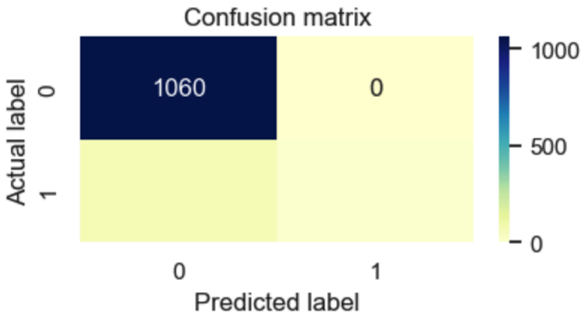
- **False Positives (FP):** 9.

- **Observation:**
  - Logistic Regression has slightly better recall (70.31%) compared to SVM but lower precision (83.33%).
  - Higher recall means it identified more churners than SVM but at the cost of more false positives (lower precision).
  - Overall, the model balances recall and precision well, making it a competitive choice.

---

3. K-Nearest Neighbors (KNN):

```
Model: knn
Accuracy: 94.76%
Precision: 100.00%
Recall: 9.23%
F1 Score: 16.90%

Classification Report:
              precision    recall  f1-score   support

           0       0.95      1.00      0.97      1060
           1       1.00      0.09      0.17        65

    accuracy                           0.95      1125
   macro avg       0.97      0.55      0.57      1125
weighted avg       0.95      0.95      0.93      1125
```



- **Accuracy:** 93.39%
- **Precision (Churn):** 100.00%
- **Recall (Churn):** 3.12%
- **F1-Score (Churn):** 6.06%
- **Confusion Matrix:**
  - **True Positives (TP):** 2.

    - **False Negatives (FN):** 62.
    - **True Negatives (TN):** 874.
    - **False Positives (FP):** 0.
- **Observation:**
    - KNN achieved perfect precision (100%) but extremely poor recall (3.12%), meaning it rarely identified actual churners.
    - The model is highly biased towards predicting the majority class (non-churners), likely due to class imbalance.
    - This makes it unsuitable for the churn prediction problem, where identifying churners is crucial.

Step 4: Cross-Validation

Cross-validation provides a more reliable estimate of model performance by splitting the data into multiple folds, ensuring that each subset of data is used for both training and validation. This method helps mitigate overfitting and provides a more accurate reflection of how well the model is expected to perform on unseen data.

Method: 5-fold cross-validation was applied to each model.

- The dataset was split into 5 subsets; 4 were used for training and 1 for testing in each fold.
- The average accuracy across folds was calculated for comparison.

The following models were evaluated using 5-fold cross-validation:

- **Logistic Regression:**
    - Average Accuracy: 97.47%
    - It achieved the strongest performance across the folds, indicating that it can generalize well to new data.
- **Support Vector Machine:**
    - Average Accuracy: 97.41%
    - The SVM model achieved the 2nd highest accuracy among the tested models. Its robustness to overfitting (especially with properly chosen kernels) makes it an excellent choice for classification

tasks.

- **K-Nearest Neighbors (KNN):**
  - Average Accuracy: 94%
  - Evaluation: While KNN performed reasonably well, its accuracy was slightly lower compared to Logistic Regression and SVM.

The code snippet:

```
from sklearn.model_selection import cross_val_score

# Function to perform cross-validation and return average accuracy
def perform_cross_validation(model, X, y, cv=5):
    cv_scores = cross_val_score(model, X, y, cv=cv)
    return cv_scores.mean()
```

```
# Cross-validation for each model
lr_cv_accuracy = perform_cross_validation(lr_model, X_scaled, y)
svm_cv_accuracy = perform_cross_validation(svm_model, X_scaled, y)
knn_cv_accuracy = perform_cross_validation(knn_model, X_scaled, y)
```

```
cv_results = {
    "Logistic Regression": lr_cv_accuracy,
    "Support Vector Machine": svm_cv_accuracy,
    "K-Nearest Neighbors": knn_cv_accuracy
}

# Convert to percentage format and print
formatted_results = {model: f"{accuracy * 100:.2f}%" for model, accuracy in cv_results.items()}
formatted_results
```

```
{'Logistic Regression': '97.47%',
 'Support Vector Machine': '97.41%',
 'K-Nearest Neighbors': '94.00%'}
```

6) Model Tuning

1. **Objective**

The goal of hyperparameter tuning is to identify the optimal set of parameters that maximize the performance of each model. This is done using GridSearchCV, which tests different combinations of hyperparameters and selects the best-performing configuration.

2. **Dataset**

   The dataset was divided into training (70%) and testing (30%) sets using `train_test_split`, ensuring a random seed (`random_state=42`) for reproducibility. Features were scaled for models sensitive to feature magnitudes, such as KNN and SVC.

3. **Methodology**

   Each model's hyperparameters were optimized using `GridSearchCV` with 5-fold cross-validation. The scoring metric for the grid search was accuracy.

   Metrics Used

   The following metrics were used for model evaluation:

   - Accuracy: Overall correctness of the predictions.
   - Precision: Proportion of correctly predicted positive cases.
   - Recall: Proportion of actual positives correctly identified.
   - F1-Score: Harmonic mean of precision and recall.
   - Confusion Matrix: Visualization of true positives, false positives, true negatives, and false negatives.

Tuning Process:

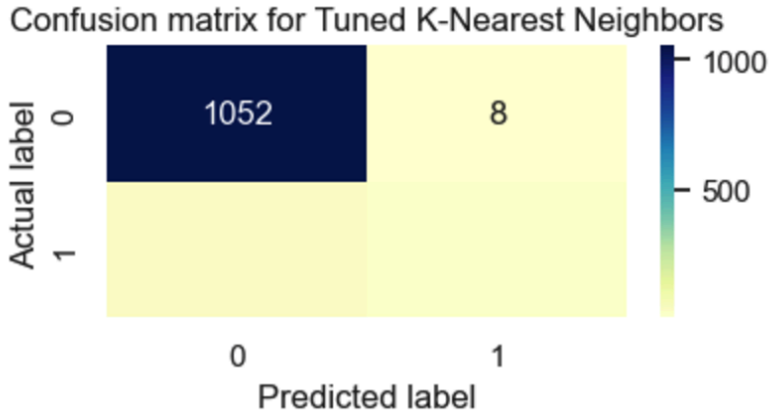
1. **K-Nearest Neighbors:**
   - **Hyperparameter Grid**: The following parameters were tested:
     - `n_neighbors`: [3, 5, 7, 9, 11, 13, 15, 17, 19, 21]
     - `weights`: ['uniform', 'distance']
     - `metric`: ['euclidean', 'manhattan', 'minkowski']
   - **Best Parameters:**
     - `metric`: 'manhattan'
     - `n_neighbors`: 5
     - `weights`: 'uniform'

   Results:

- **Accuracy:** 96%
    - **Class 0 (Majority Class):**
        - Precision: 0.97, Recall: 0.99, F1-Score: 0.98
    - **Class 1 (Minority Class):**
        - Precision: 0.77, Recall: 0.42, F1-Score: 0.54
    - **Macro Avg (0 and 1):**
        - F1-Score: 0.76

Insights: While the KNN model performed well for the majority class (class 0), it struggled with the minority class (class 1), resulting in low recall and F1-scores for the minority class.

```
Classification Report for Tuned K-Nearest Neighbors:
              precision    recall  f1-score   support

           0       0.97      0.99      0.98      1060
           1       0.77      0.42      0.54        65

    accuracy                           0.96      1125
   macro avg       0.87      0.70      0.76      1125
weighted avg       0.95      0.96      0.95      1125
```



2. **Logistic Regression:**
    - **Hyperparameter Grid**: The following parameters were tested:
        - `penalty`: ['l1', 'l2']
        - `C`: [0.01, 0.1, 1, 10, 100]
        - `solver`: ['lbfgs', 'liblinear']
        - `max_iter`: [500, 1000, 2000]

    Best Parameters:

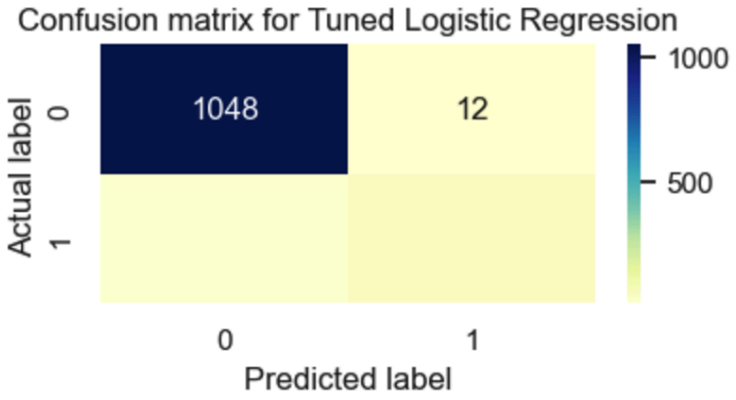
        - `C`: 1
        - `max_iter`: 500
        - `penalty`: 'l1'
        - `solver`: 'liblinear'

    Results:

        - **Accuracy:** 97%
            - **Class 0 (Majority Class):**
                - Precision: 0.98, Recall: 0.99, F1-Score: 0.99
            - **Class 1 (Minority Class):**
                - Precision: 0.80, Recall: 0.72, F1-Score: 0.76
            - **Macro Avg (0 and 1):**
                - F1-Score: 0.87

    Insights: Logistic Regression performed slightly worse than SVC in terms of minority class metrics, though it achieved a solid performance overall. Its simplicity makes it a good baseline for comparison.

```
Best Parameters for Logistic Regression:  {'C': 1, 'max_
Classification Report for Logistic Regression:
              precision    recall  f1-score   support

           0       0.98      0.99      0.99      1060
           1       0.80      0.72      0.76        65

    accuracy                           0.97      1125
   macro avg       0.89      0.86      0.87      1125
weighted avg       0.97      0.97      0.97      1125
```



3. Support Vector Machine (SVM)

Hyperparameter Grid: The following parameters were tested:

- `C`: [0.1, 1, 10, 100]
- `gamma`: [1, 0.1, 0.01, 0.001, 0.0001]
- `kernel`: ['rbf']

Best Parameters:

- `C`: 10
- `gamma`: 0.01
- `kernel`: 'rbf'

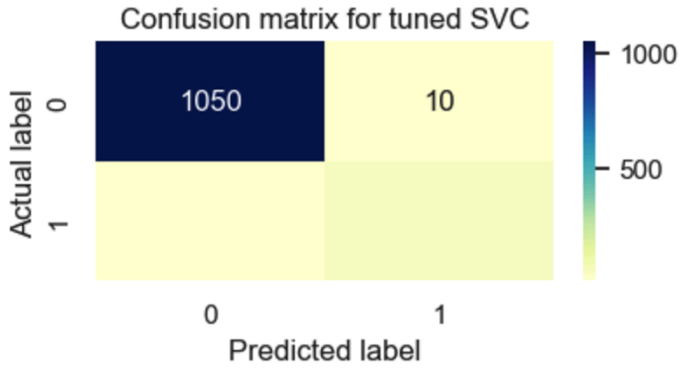
Results:

- **Accuracy:** 98%

- **Class 0 (Majority Class):**
    - Precision: 0.99, Recall: 0.99, F1-Score: 0.99
- **Class 1 (Minority Class):**
    - Precision: 0.84, Recall: 0.78, F1-Score: 0.81
- **Macro Avg (0 and 1):**
    - F1-Score: 0.90

- **Insights:** The SVC model demonstrated a more balanced performance across both classes. Its high F1-score for the minority class (0.81) shows its effectiveness in handling imbalanced datasets. The RBF kernel was instrumental in achieving this performance.

```
Best Parameters for SVC:  {'C': 10, 'gamma': 0.01, 'kernel': 'rbf'}
              precision    recall  f1-score   support

           0       0.99      0.99      0.99      1060
           1       0.84      0.78      0.81        65

    accuracy                           0.98      1125
   macro avg       0.91      0.89      0.90      1125
weighted avg       0.98      0.98      0.98      1125
```



- **Chosen Model:** Tuned SVM
    - **Reason:** The **Support Vector Classifier (SVC)** emerged as the best model, offering the highest performance for both classes, especially the minority class.
    - The SVM model is thus chosen as the final model for deployment in

predicting customer churn.

7) Model Interpretation

In this section, we utilize the Feature Importance score and SHAP (Shapley Additive explanations) to interpret the predictions made by our best-performing model, the Support Vector Machine (SVM).

1. Feature Importance

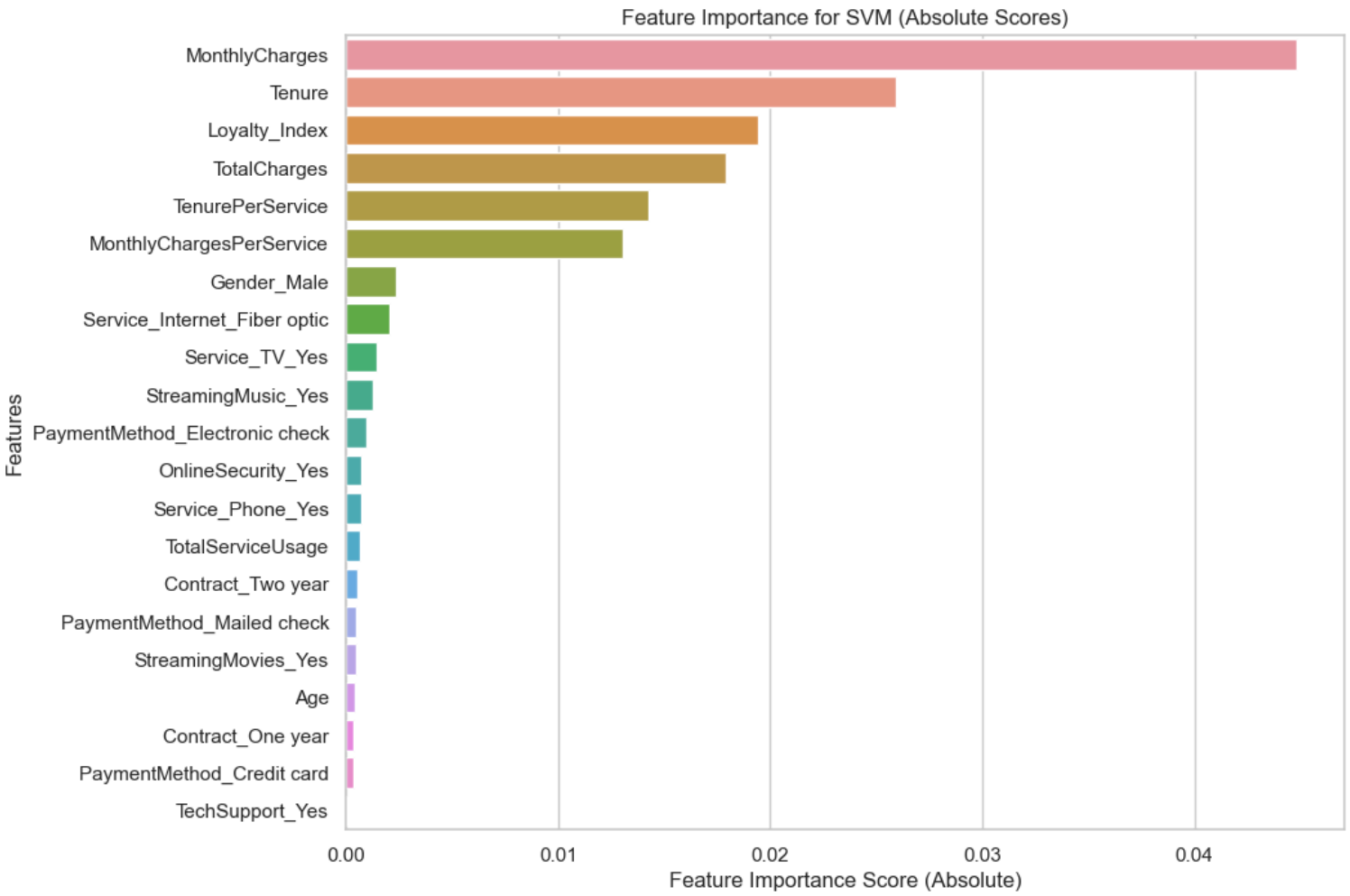
Objective:

To understand the contribution of each feature to the predictive performance of the SVM model by measuring the impact on accuracy when the values of a single feature are randomly shuffled.

Steps:

1. **Compute Permutation Importance**:
   - Used `permutation_importance` from `sklearn.inspection`.
   - Ran 30 random shuffles (`n_repeats=30`) for each feature to calculate the mean decrease in accuracy when the feature values are permuted.
2. **Extract Importance Scores**:
   - Calculated the absolute mean importance scores for each feature.
   - Linked feature names to importance scores.
3. **Visualization**:
   - Created a sorted data frame of feature importance scores.
   - Visualized the importance using a horizontal bar plot for better interpretability.

Top Features (in order):



Detailed SHAP Summary Plot Interpretation:

The SHAP (SHapley Additive exPlanations) summary plot provides a comprehensive understanding of how each feature contributes to the SVM model's predictions. By analyzing SHAP values, we gain insights into the direction and magnitude of the influence each feature has on the model's output, helping us identify the key drivers of customer churn.

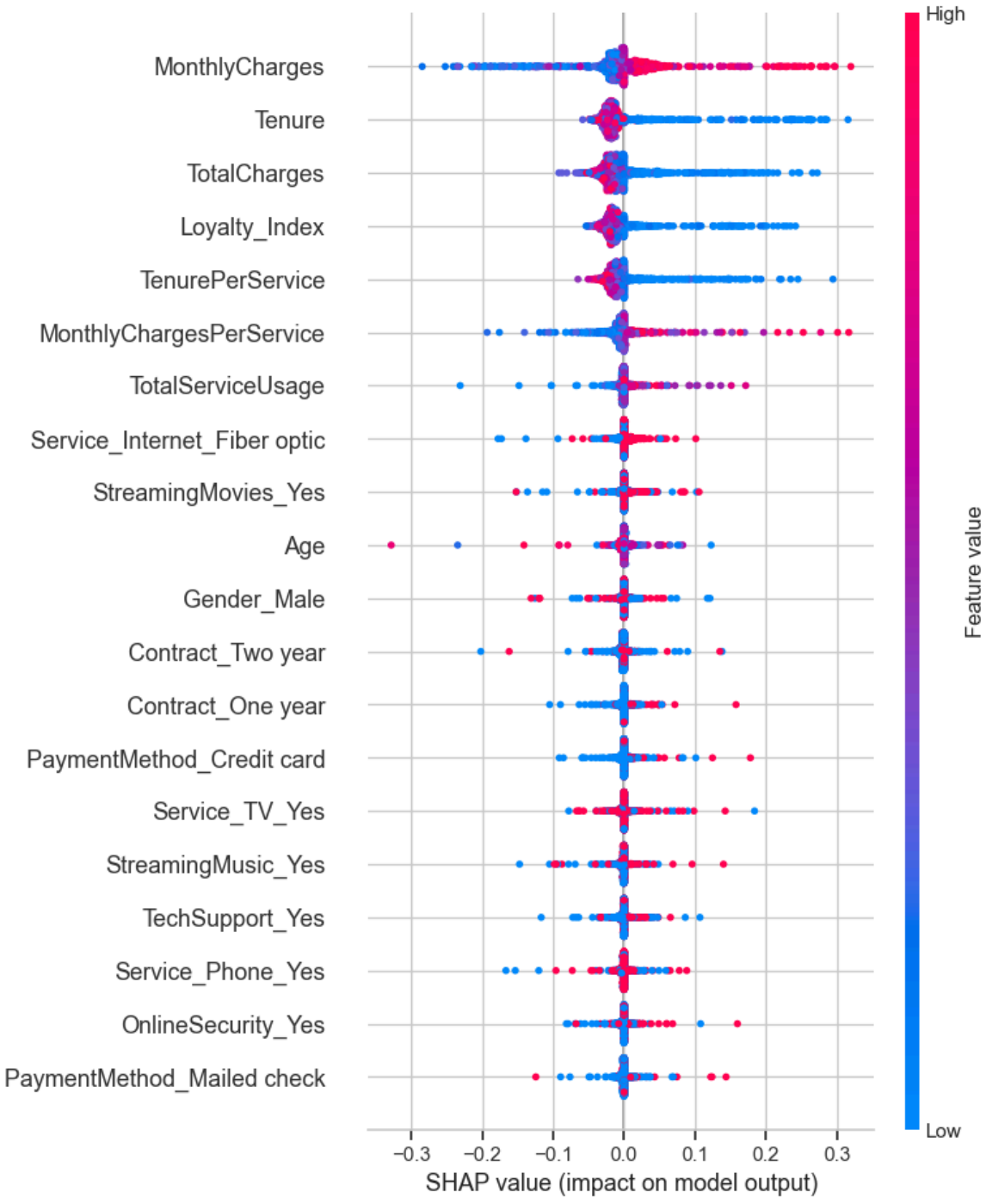
Objective:

To provide detailed insights into how each feature contributes to individual predictions by measuring the marginal contribution of each feature to the model's output.

Steps:

1. **KernelExplainer**:
    - Used shap.KernelExplainer for the SVM model as it is a non-linear model.
    - Calculated SHAP values for a subset of the test data (nsamples=100) to balance computational cost and accuracy.
2. **SHAP Summary Plot**:
    - Plotted SHAP values to visualize the impact of each feature on the model's predictions.
    - Features were ranked by their average absolute SHAP value, with colour representing the feature value (e.g., red for high and blue for low).

Results:


MonthlyCharges
Tenure
TotalCharges
Loyalty_Index
TenurePerService
MonthlyChargesPerService
TotalServiceUsage
Service_Internet_Fiber optic
StreamingMovies_Yes
Age
Gender_Male
Contract_Two year
Contract_One year
PaymentMethod_Credit card
Service_TV_Yes
StreamingMusic_Yes
TechSupport_Yes
Service_Phone_Yes
OnlineSecurity_Yes
PaymentMethod_Mailed check
−0.3
−0.2
−0.1
0.0
0.1
0.2
0.3
SHAP value (impact on model output)
High
Feature value
Low

Feature Analysis

1. MonthlyCharges

- **Observation**:
  - Customers with high MonthlyCharges exhibit significant churn likelihood, with SHAP values between 0.1 and 0.3.
  - Customers with low MonthlyCharges are less likely to churn, as evidenced by negative SHAP values.
- **Insight**:
  - Higher monthly costs increase dissatisfaction, likely reflecting either perceived poor value or cost sensitivity.

---

2. Tenure

- **Observation**:
  - High-tenure customers have strong negative SHAP values (-0.2 to -0.3), indicating low churn risk.
  - Customers with low tenure contribute to churn predictions, as shown by SHAP values > 0.1.
- **Insight**:
  - Newer customers haven't yet developed loyalty and are more prone to leaving if their expectations aren't met during onboarding or early interactions.

---

3. TotalCharges

- **Observation**:
  - Low TotalCharges correlates with churn, with SHAP values > 0.1, while high TotalCharges reduces churn likelihood.
- **Insight**:
  - Customers with low total payments likely have shorter tenure or

minimal engagement, making them less invested in the service and easier to lose.

---

4. Loyalty_Index (Interaction Term)

- **Description**: An engineered feature representing the combined effect of TotalServiceUsage and Tenure.
  - Formula: Loyalty_Index = TotalServiceUsage * Tenure
- **Observation**:
  - Low loyalty scores strongly predict churn (SHAP values > 0.1).
  - High scores indicate strong retention (SHAP values between -0.1 and -0.3).
- **Insight**:
  - Customers who are both long-term and use multiple services tend to be loyal. Those with limited engagement across both dimensions are at high risk of leaving.

---

5. TenurePerService (Interaction Term)

- **Description**: The average tenure across the number of services a customer uses.
  - Formula: TenurePerService = Tenure / TotalServiceUsage
- **Observation**:
  - High values correspond to retention (negative SHAP values), while low values are linked to churn (positive SHAP values).
- **Insight**:
  - Customers who stay long while using multiple services tend to be more satisfied. Conversely, single-service users or recent customers are at higher churn risk.

---

6. MonthlyChargesPerService (Interaction Term)

- **Description**: The average monthly charge per service.
  - Formula: MonthlyChargesPerService = MonthlyCharges / TotalServiceUsage
- **Observation**:
  - High values (red) are directly tied to churn, with SHAP values > 0.1, indicating that high relative costs lead to dissatisfaction.
  - Low values (blue) contribute to retention.
- **Insight**:
  - Customers paying disproportionately high costs per service are more likely to churn, reflecting value sensitivity.

---

7. TotalServiceUsage (Interaction Term)

- **Observation**:
  - High service usage showing churn risk (possibly due to dissatisfaction or unmet expectations) and others displaying retention.
- **Insight**:
  - Underutilization of services signals disengagement, while overuse may result in dissatisfaction, especially if expectations are not met.

8) Final Conclusion and Opinions

Pre-Exploration Phase

The project began with an in-depth exploration of the dataset. Summary statistics, visualizations like histograms and scatter plots, and correlation analyses provided a clear understanding of the data distribution and relationships among features. These insights laid the groundwork for effective data preprocessing.

Data Preprocessing Phase

Key preprocessing steps included:

1. **Handling Missing Data:** Missing values in numerical and categorical features were imputed using the median and mode, respectively.
2. **Outlier Treatment:** Significant outliers in features like MonthlyCharges and TotalCharges were capped using the IQR method.
3. **Feature Scaling and Encoding:** Numerical features were normalized, and categorical features were converted into numerical representations using one-hot encoding.

The cleaned dataset was saved and used for detailed exploratory data analysis.

In-Depth Exploration with the Cleaned Dataset

I conducted a detailed exploratory data analysis (EDA) on the preprocessed dataset:

- **Key Observations:**
  - **Tenure:** Shorter tenures were strongly associated with higher churn rates, indicating the need for better onboarding processes.
  - **MonthlyCharges:** Customers with higher monthly charges were more likely to churn, suggesting cost concerns.
  - **TotalCharges:** Low total charges correlated with churn, often due to

shorter tenures and limited service use.

These findings guided the feature engineering and modeling processes.

Feature Engineering Phase

New features were engineered to capture deeper insights:

1. **Loyalty Index:** Combined tenure and service usage to reflect customer engagement.
2. **MonthlyCharges Per Service:** Highlighted customers paying disproportionately higher costs per service.
3. **Tenure Per Service:** Provided insights into customer satisfaction and loyalty.

These features enhanced the model's interpretability and predictive power, emphasizing the significance of multi-service engagement and cost-effectiveness.

Model Training and Hyperparameter Tuning

I developed and evaluated several models, including Support Vector Machines (SVM), Logistic Regression, and K-Nearest Neighbors (KNN). SVM emerged as the best-performing model:

- **SVM Results:**
    - Accuracy: 98%
    - Precision (Churn): 84%
    - Recall (Churn): 78%
    - F1-Score (Churn): 81%

Hyperparameter tuning further improved performance. The SVM model was optimized using parameters such as C, gamma, and kernel, achieving a balanced performance across both churners and non-churners.

Model Interpretation Phase

Using SHAP (SHapley Additive ExPlanations), I interpreted the SVM model to identify the most influential features.

Actionable Insights Based on Feature Analysis

1. **MonthlyCharges:**
    - **Action:** Segment customers with high monthly charges and offer personalized discounts or alternative pricing tiers to address cost sensitivity. For example, introduce a "value pack" bundle that consolidates services at a lower overall cost.
    - **Implementation:** Regularly analyze billing data to identify customers in the upper quartile of monthly charges and target them with campaigns emphasizing cost savings.
2. **Tenure:**
    - **Action:** Develop a program aimed at new customers (low tenure). This program could include tutorials, welcome discounts, or personalized setup assistance.
    - **Implementation:** Automate customer communication in the first 3-6 months, using email or SMS to share usage tips, value demonstrations, and exclusive offers to improve early satisfaction.
3. **TotalCharges:**
    - **Action:** Engage customers with low total charges by promoting additional services through upselling or cross-selling. For instance, offer a free trial of premium features like streaming or enhanced security.
    - **Implementation:** Use customer service teams or automated marketing tools to suggest additional services to customers, especially as they approach the end of their first contract.
4. **Loyalty Index:**
    - **Action:** Reward loyal customers with exclusive perks, such as discounts, priority support, or early access to new features, to show appreciation and reinforce their commitment.

- **Implementation:** Create a tiered loyalty program based on the Loyalty Index, with escalating rewards tied to tenure and service usage levels.

5. **TenurePerService:**
   - **Action:** Focus on customers with low tenure per service by promoting multi-service adoption. Highlight cost savings and convenience when bundling services.
   - **Implementation:** Use targeted ads or direct outreach to encourage single-service users to upgrade to a bundle, offering discounts for multi-service plans.
6. **MonthlyChargesPerService:**
   - **Action:** Reduce churn for customers with high monthly charges per service by bundling additional services or introducing lower-cost alternatives. Showcase how bundling provides better value.
   - **Implementation:** Proactively identify high-risk customers using this metric and initiate calls or campaigns with specific service bundles that reduce the cost per service.
7. **TotalServiceUsage:**
   - **Action:** Address both underutilization and overutilization:
     - **Underutilization:** Reach out to customers who use only a small subset of their available services and offer tutorials or guides to help them maximize service benefits.
     - **Overutilization:** Provide overusing customers with feature enhancements or add-ons that enhance their experience and prevent dissatisfaction.
   - **Implementation:** Monitor usage patterns and deploy automated systems to flag under-engaged customers for intervention or reach out with add-on offers for high-engagement users.

Strategic Recommendations

1. **Customer Education:**
    - Conduct workshops or create content (videos, webinars) to educate customers on how to maximize value from their subscriptions, targeting those with underutilization.
2. **Proactive Retention Strategies:**
    - Use the Loyalty Index and TenurePerService metrics to create predictive churn alerts, enabling the team to proactively reach out to at-risk customers with tailored solutions.
3. **Dynamic Pricing:**
    - Implement dynamic pricing based on MonthlyChargesPerService and offer flexible options to high-risk customers, ensuring affordability without compromising perceived value.
4. **Enhanced Personalization:**
    - Use machine learning models to continuously update customer profiles, adjusting recommendations and campaigns dynamically to reflect usage and risk patterns.
5. **Service Feedback Loop:**
    - For customers showing churn risk due to dissatisfaction (e.g., high usage but churn), implement quick feedback loops like surveys or real-time chat to understand and resolve pain points.