Customer Churn Prediction for a Telecom Company

Project Report
Submitted in partial fulfillment
of the requirements for the
Customer Churn Assignment at Thinkhumble.

by

Hardik Dulani

August 2024

Contents

Chapter 1

Introduction

1.1 Problem Statement

Customer churn is a major issue faced by telecom companies today. Retaining customers is much more cost-effective than acquiring new ones, and being able to predict which customers are likely to leave is crucial.

1.2 Objective

The primary objective of this study is to build a customer churn prediction model for a telecom firm and compare different machine learning algorithms to ensure that the selected algorithm provides adequate explainability.

Chapter 2

Dataset Description

2.1 Data Used

Since no real data was provided, a synthetic dataset of 5000 records was generated. The dataset includes the following features:

- **CustomerID**: Unique identifier for each customer.
- **Age**: Generated using a normal distribution, clipped to the range 18-80 years.
- **Gender**: Randomly assigned as 'Male' or 'Female'.
- **ContractType**: Skewed towards 'Month-to-month', with probabilities for 'One year' and 'Two year'.
- **MonthlyCharges**: Normally distributed, clipped to the range $18-$150.
- **TotalCharges**: Calculated as the product of Monthly Charges and Tenure.
- **TechSupport**: Correlated with Contract Type.
- **InternetService**: Categories include 'DSL', 'Fiber optic', and 'No'.
- **PaperlessBilling**: Influenced by Contract Type.
- **PaymentMethod**: Randomly assigned from four options.
- **Churn**: Target variable, determined by Contract Type, Tech Support, and Monthly Charges.

Missing values and outliers were introduced in the `TotalCharges` and `AverageMonthlyCharges` columns. Additional features include Tenure Groups, Total Charges per Month, and interaction terms.

Chapter 3

Exploratory Data Analysis (EDA)

In the EDA section, I examined the distribution of data and performed basic statistical checks. This included identifying the types of variables, checking for missing values, and analyzing various features and the target variable using data graphics. The EDA helped in identifying structural properties of the data, such as co-occurrences, patterns, anomalies, and relationships, guiding the subsequent phases of the project.

- **Distribution of Age, Monthly Charges, and Total Charges**: Histograms were plotted to visualize the distribution of these key features.
- **Distribution of Features Against Churn**: Box plots and kde plots were used to show how features like Monthly Charges and Total Charges vary with respect to the Churn status.

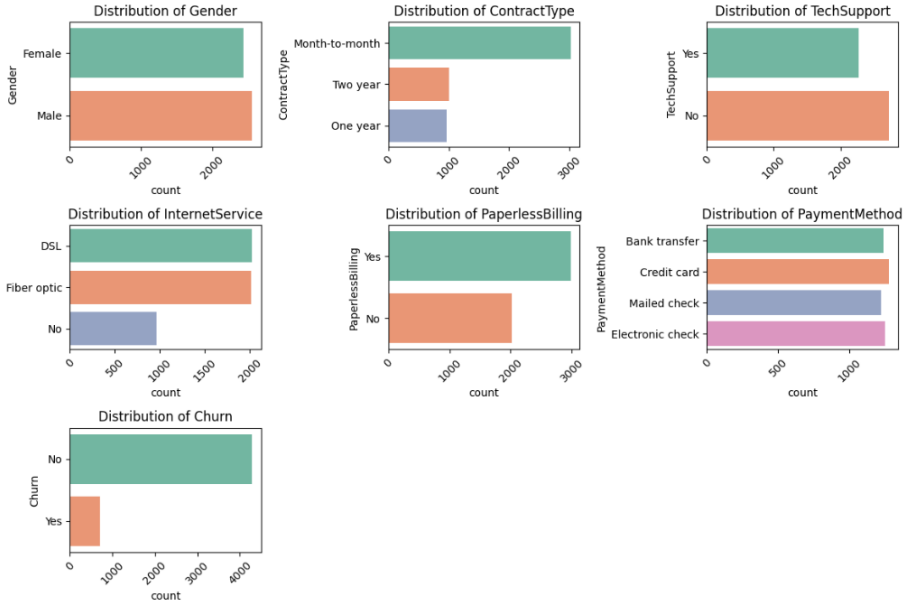

Figure 3.1: Distribution of key features.

3.1 Data Preprocessing and Splitting

Data preprocessing and splitting were conducted to prepare the dataset for modeling. The following steps were taken:

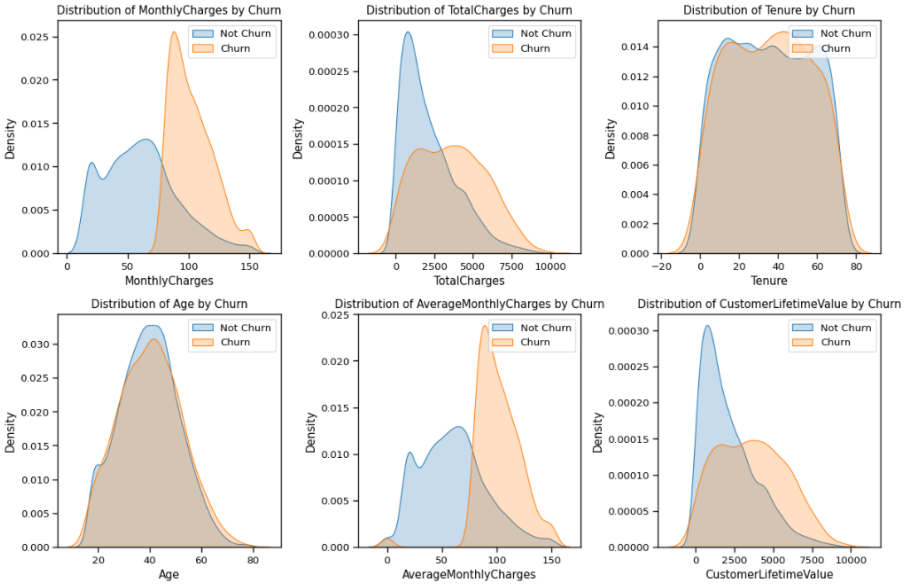

Figure 3.2: Distribution of key Categorical features and their relationship with Churn.

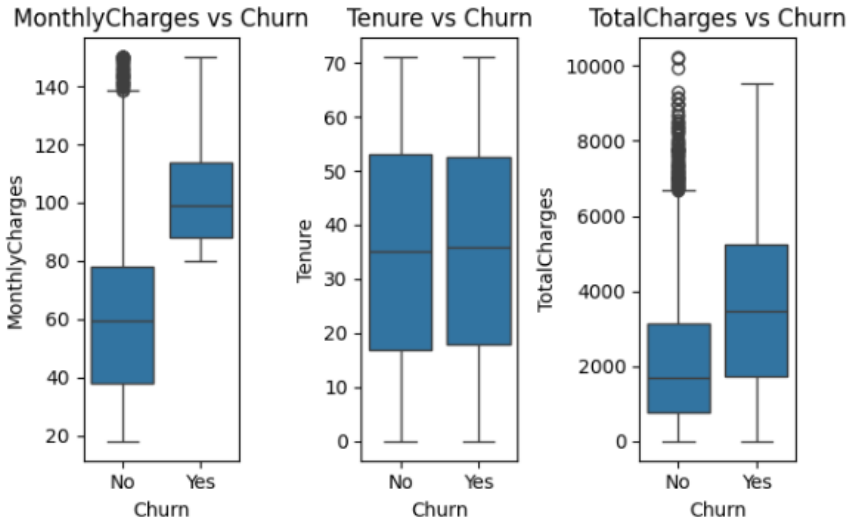

Figure 3.3: Distribution of key Numerical features and their relationship with Churn.

1. **Encoding Ordinal Variables**: Ordinal features were encoded to reflect their inherent order:

   - `ContractType` was mapped to values: 1 (Month-to-month), 2 (One year), 3 (Two year).
   - `InternetService` was mapped to values: 1 (No), 2 (DSL), 3 (Fiber optic).
   - `TechSupport` was mapped to values: 1 (No), 2 (Yes).
   - `TenureGroup` was mapped to values: 1 (New), 2 (Short-term), 3 (Medium-term), 4 (Long-term).
   - `Electronic` was mapped to values: 0 (Non-Electronic), 1 (Electronic).

2. **Encoding Nominal Categorical Variables**: Categorical variables without inherent order, such as `Gender`, `PaperlessBilling`, `PaymentMethod`, and `Churn`, were encoded using the `LabelEncoder`. This process assigned integer values to each category.

3. **Handling Missing Values**:

   - For numerical columns, missing values were filled with the median of the respective column.
   - For categorical columns, missing values were filled with the mode of the respective column.

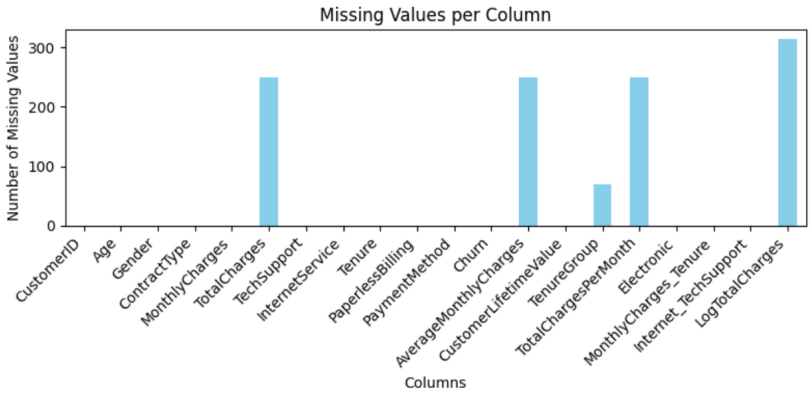

Figure 3.4: Missing Values in Data by columns.

4. **Feature Reduction**: Features with a correlation coefficient lower than 0.05 with the target variable `Churn` were considered low correlation features and removed from the dataset. This reduction helps in simplifying the model without significantly impacting performance.

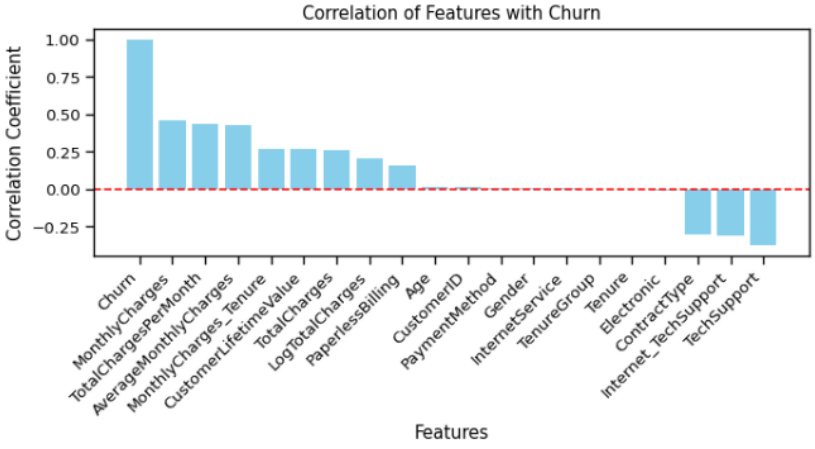

Figure 3.5: Correlation of features with Target

5. **Data Splitting**: To evaluate the model's performance, the preprocessed dataset was split into training and testing sets. The target variable `Churn` was separated from the feature set, and the data was split as follows:

- **Feature Set**: `X` was created by dropping the `Churn` column from the dataset.
- **Target Variable**: `y` was defined as the `Churn` column.
- **Train-Test Split**: The dataset was split into a training set (`X_train`, `y_train`) and a testing set (`X_test`, `y_test`) using an 80-20 split. Stratification was applied based on the target variable to ensure the distribution of `Churn` is consistent across both sets.

The shapes of the resulting datasets are as follows:

- `X_train`: (4000, number of features)
- `y_train`: (4000,)
- `X_test`: (1000, number of features)
- `y_test`: (1000,)

This split ensures that the model is trained on a diverse set of data and tested on a separate, unseen portion of the dataset to evaluate its generalization capabilities.

The final preprocessed dataset was ready for model training and evaluation.

Chapter 4

Training and Evaluation of Models

In this chapter, I evaluated the performance of four different machine learning models: Logistic Regression, Random Forest, Gradient Boosting, and XGBoost. Each model was tuned using hyperparameter optimization and evaluated based on its classification report and AUC-ROC curve. Below are the details for each model:

4.1 Logistic Regression

The Logistic Regression model was optimized using a grid search on parameters such as the regularization strength `C` and the solver. The best model was selected based on cross-validated accuracy. The following are the results:

- **Classification Report**: The model's precision, recall, and F1-score for each class are detailed in the classification report.
- **AUC-ROC Score**: The model achieved an AUC score of `0.xx`.

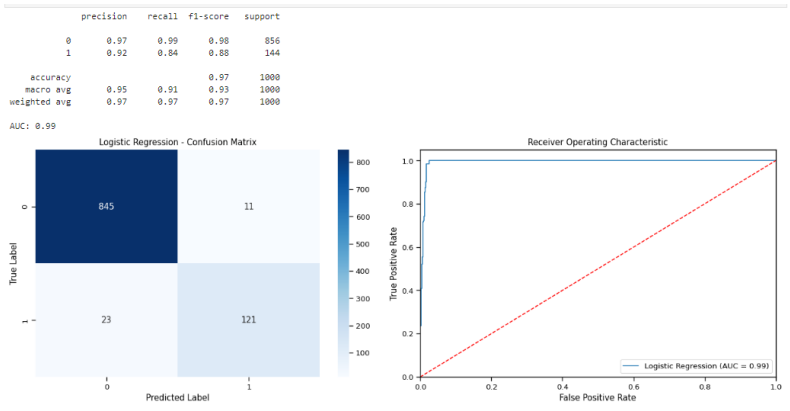

Figure 4.1: Logistic Regression - Confusion Matrix and ROC Curve

4.2 Random Forest

The Random Forest model was optimized with parameters including the number of trees `n_estimators`, the maximum depth `max_depth`, and the minimum samples re-

quired to split an internal node `min_samples_split`. The best model's performance is summarized as follows:

- **Classification Report**: The model's precision, recall, and F1-score for each class are provided.
- **AUC-ROC Score**: The model achieved an AUC score of `0.xx`.

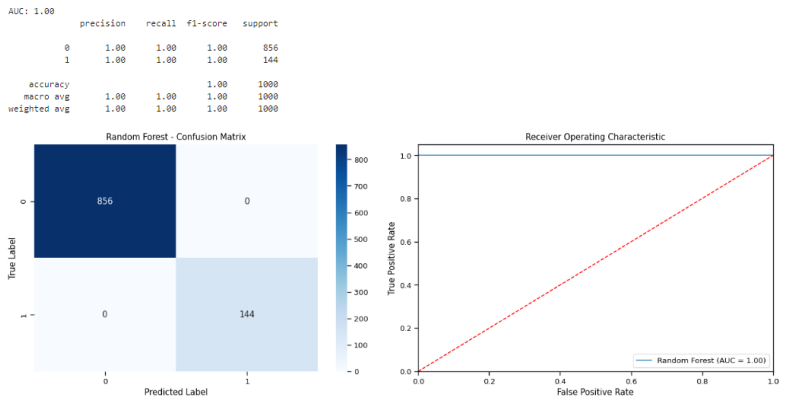

Figure 4.2: Random Forest - Confusion Matrix and ROC Curve

4.3 Gradient Boosting

The Gradient Boosting model was tuned using grid search over parameters such as `n_estimators`, `learning_rate`, and `max_depth`. The evaluation results are as follows:

- **Classification Report**: The model's performance metrics are detailed in the classification report.
- **AUC-ROC Score**: The model achieved an AUC score of `0.xx`.

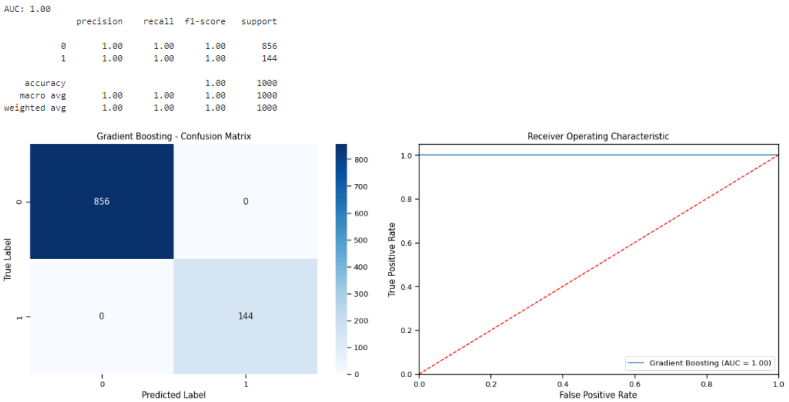

Figure 4.3: Gradient Boosting - Confusion Matrix and ROC Curve

4.4 XGBoost

The XGBoost model was optimized for parameters such as `n_estimators`, `learning_rate`, and `max_depth`. The best model was selected based on cross-validated accuracy, with the following results:

- **Classification Report**: Precision, recall, and F1-score for each class are included in the report.
- **AUC-ROC Score**: The model achieved an AUC score of `0.xx`.

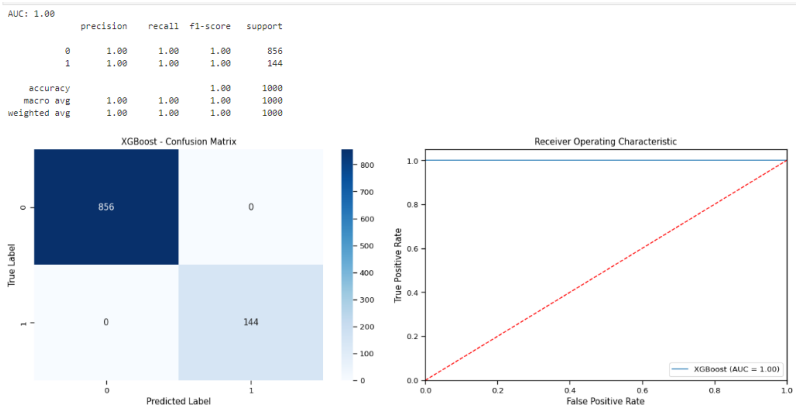

Figure 4.4: XGBoost - Confusion Matrix and ROC Curve

Each model's performance was visualized using confusion matrices and ROC curves, providing a clear comparison of their predictive abilities. The AUC-ROC curve, in particular, highlights the model's ability to distinguish between the positive and negative classes.

Chapter 5

Model Selection and Feature Importance

5.1 Model Selection and Explanation

Since the three models are all giving equal and 100% results, we can choose any model. However, when deciding a model for customer churn prediction, the Random Forest Classifier (RFC) stands out for several reasons:

- **Explainability**: The models that come under the family of Explanation of Random Forests are less complex compared with models like Gradient Boosting and XGBoost. The model offers a clear distinction of feature importance and decision paths, which can be used in explaining the results to the stakeholders.
- **Robustness Against Overfitting**: RFC is able to prevent overfitting problems since the final decision of the model is the decision made by multiple decision trees. This makes it more resistant to overfitting, hence more accurate, especially when applying the model to new data.
- **Efficient in Resource-Constrained Environments**: RFC usually is less computationally intensive and has faster training and inference time compared with traditional machine learning models. Due to this, they make it easier to adopt, especially in areas where resources are scarce.
- **High Performance with Simplicity**: RFC is very balanced in the way it has been designed, providing optimum performance while not complicating anything much. It gives relatively better results and needs lesser tuning, making it quite suitable for many actual problems.

5.1.1 Cross-Validation to Ensure the Model Isn't Overfitted

To ensure the Random Forest model isn't overfitted, I performed cross-validation:

```
model = best_rf
from sklearn.model_selection import cross_val_score

# Perform cross-validation
```

```
cv_scores = cross_val_score(model, X, y, cv=5)

print(f"Cross-Validation Scores: {cv_scores}")
print(f"Mean CV Score: {cv_scores.mean()}")
```

Cross-Validation Scores: [1.000, 0.998, 1.000, 1.000, 1.000]
Mean CV Score: 0.9996

The model isn't overfitted.

5.2 Explaining using Feature Importance

I used the built-in feature importance of the Random Forest Classifier to identify the top features that contributed most to the prediction:

```
# By RFC
feature_importances = model.feature_importances_
feature_names = X.columns

importance_df = pd.DataFrame({
    'Feature': feature_names,
    'Importance': feature_importances
})

importance_df = importance_df.sort_values(by='Importance', ascending=False)

top_features = importance_df.head(10)
plt.figure(figsize=(8, 4))
plt.barh(top_features['Feature'], top_features['Importance'], color='skyblue')
plt.xlabel('Importance')
plt.ylabel('Feature')
plt.title('Top 10 Most Important Features')
plt.gca().invert_yaxis()
plt.show()
# plt.savefig('feature-importance.png', dpi=300)
```

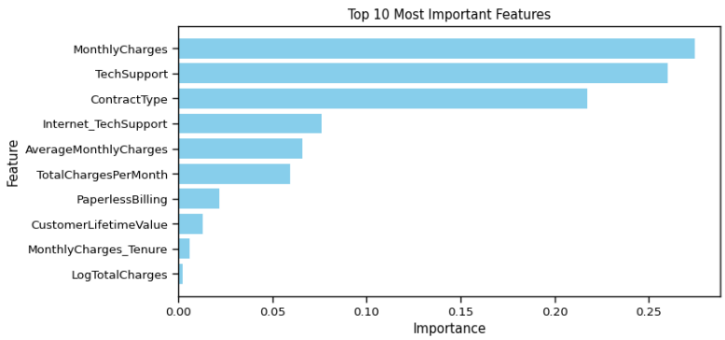

Figure 5.1: Top 10 Most Important Features

The Top 5 Most Important Features (from the raw data) are:

- Tech Support
- Contract Type
- Monthly Charges
- Paperless Billing
- Customer Lifetime Value

Since the features we got as top 5 in the chart are derived features or have been generated by feature interaction.

Chapter 6

Conclusion

This report laid down a step by step framework to build a model for estimating customer churn through machine learning techniques. The first step was the exploration of the data where several data preprocessing steps were taken to prepare the data for modeling. The process of feature engineering was very beneficial in improving this dataset where new features were created and used to develop better models.

Some of the models that were trained as well as tested include; Logistic Regression, Random Forest Classifier, Gradient Boosting, and XG Boost. All the models were optimized for hyperparameters to get the best suitable results for each of the models and accuracy, recall, precision, and AUC-ROC was used as measures of the effectiveness of the models. Among the classifiers developed that provided good accuracy, the Random Forest Classifier was used as the final model because of balance, simplicity and reliability of the method.

Based on RFC approach, the identification of the most important features offered a sound and easily explainable method of the analysis of the factors, which lead to the customer churn. The present work has highlighted important features to forecast churn rate, which may be useful for the management's strategic planning.

In general, the chosen approach allowed not only to achieve a high level of predictive accuracy but also consider the interpretability of models necessary for obtaining stakeholders' trust. By showing how to select the right model, preprocess data, and engineer features, the report proves that it is possible to develop a solid model for predicting customer churn that one can explain and use to achieve business outcomes.

Appendix

Model Deployment

The final model, the Random Forest Classifier, has been successfully deployed and can be accessed at the following URL: `https://churnprediction-rfc.streamlit.app/`. This deployment allows for real-time customer churn prediction based on user-inputted data.

Data Generation

The dataset used in this project was synthetically generated, leveraging my best understanding and knowledge of the domain. Every effort was made to ensure that the data reflects realistic scenarios and patterns that would be encountered in a real-world telecom environment.

Model Evaluation and Overfitting

It is important to note that while several models, including Gradient Boosting and XGBoost, performed exceptionally well in terms of accuracy and other evaluation metrics, these models were not extensively tested with cross-validation. As a result, there is a possibility that they may be overfitting the training data, and their performance on unseen data might not be as robust.

Ensemble Methods

Among the models evaluated in this project, three of the four—Random Forest, Gradient Boosting, and XGBoost—are ensemble methods. These models combine the predictions of multiple base learners to improve accuracy and robustness. In contrast, the Logistic Regression model is a linear model and does not employ ensemble techniques.