THOMPSON RIVERS UNIVERSITY

Assessing and Implementing Supervised Machine Learning for Effective Customer Churn Prediction

By

Khanh Van Tran

A GRADUATE PROJECT REPORT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Science in Data Science

KAMLOOPS, BRITISH COLUMBIA

April, 2024

SUPERVISOR

Dr. Jabed Tomal

Dr. Md Erfanul Hoque

ABSTRACT

This study has been motivated to address a very critical problem faced in the telecom industry: customer churn. The research goal is not only to model supervised machine learning on the telecom company dataset to predict churn customers but also to understand key factors that cause customer churn. The data for this study has been taken from 7043 customers of a telecom company based in California. The analysis has been performed on various machine learning algorithms, such as standard ones like logistic regression, support vector machines, and complex ensemble methods like LightGBM and XGBoost. This study has measured machine learning algorithms' performance using accuracy, precession, recall, and F1 score metrics. The top three performers on this telecom churn dataset were LightGBM, Gradient Boosting, and AdaBoost. The LightGBM classifier got the best scores across all measures, with scores of 0.89 for accuracy, 0.84 for precession, 0.76 for recall, and 0.80 for F1. The research offers valuable insights into the primary aspects that contribute to churn, including the duration of customer retention, the types of contracts, the number of referrals, and the amount of invoicing. This contributes to the company's comprehension of consumer behavior and assists in the creation of focused retention efforts. Additionally, the LightGBM classifier has been implemented in the live Azure Cloud for real-time prediction, which enables firms to promptly identify potential customers and take proactive actions to keep them.

Key Words: Machine learning; Churn prediction; Ensemble methods; Random Forests s; K-nearest neighbors, Support vector machine, Logistic regression.

ACKNOWLEDGEMENTS

Firstly, I would like to express my sincere gratitude to my two supervisors, Dr. Jabed Tomal and Dr. Md Erfanul Hoque for teaching, supervising, guiding, and mentoring me in the entire process of my graduation project, which indeed has helped me to shape my research properly and overcome lots of obstacles.

An additional warm thank you goes to the instructors in the MSc Data Science program at Thompson Rivers University: Dr. Roger Yu, Dr. Mohamed Tawhid, Dr. Becky Wei Lin, Dr. Mateen Shaikh, and Dr. Mila Kwiatkowska.

And I'd like to thank my family for the love and support they've shown me every step of the way. Their unwavering faith in me helped carry me through all this.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview of Customer Churn

Customer churn is also known as customer attrition. It is a situation wherein a customer stops purchasing products and services from a company. Customers are hard to acquire and might be very expensive for an organization. If a customer does not use your firm's products and services in the long run, it will inevitably lead to the bad fate of an organization. This is why customer retention is also as important as customer acquisition. Without appropriate attention to customer churn management, it can result in heavy monetary losses, reduced profitability, and loss of visibility in the market (Jain et al., 2020). This is why organizations started relying on predictive analytics or machine learning approaches to predict or avoid customer churn.

Since corporations now have access to enormous amounts of customer data through large-scale databases and digital platforms, such information as consumer profiles, transaction histories, notes from communication with customer service personnel, and their interactions with the internet. Companies can now use this vast amount of data to create effective machine learning models for predicting customer attrition. Business entities can lower their customer attrition rate by using previously logged cases to identify trends and markers that predict their churn rates (Khodabandehlou and Zivari Rahman, 2017).

As a result, an impressive number of studies tell us that machine learning models can effectively predict churn for different industries such as telecom, subscription-based services, e-commerce, and financial services (Geiler et al., 2022, Jain et al., 2020, Lalwani et al., 2022, Qureshi et al., 2013, Rahman and Kumar, 2020, Ullah et al., 2019). These models can provide valuable customer insights and are particularly useful for companies to personalize their marketing efforts, increase customer experience, and execute focused retention strategies. Some of the impacts on outcomes can be quite dramatic, and using machine learning on churn can yield great returns on investment.

1.2 Project Objectives

As highlighted above, it is essential to understand why churn occurs and make plans to reduce it. Therefore, it increases customer loyalty and achieves corporate objectives. The following are the main goals of this study:

1. Identifying the primary factors contributing to customer churn: This research aims to figure out the main factors that lead to consumers terminating the service by analyzing customer data and identifying trends. Being able to understand these characteristics facilitates the design of targeted interventions aimed at addressing the factors behind churn.

2. Investigating the correlation between different characteristics and turnover status will offer valuable insights into consumer behavior and preferences. The research will provide methods to boost client retention by concentrating on individualized products and enhanced service quality.

3. Create a dependable machine learning model for predicting customer churn. The project attempts to develop a prediction model using sophisticated machine learning techniques to reliably anticipate client attrition. The project aims to provide

a strong tool for proactive churn control by evaluating different algorithms and enhancing model performance.

To address these research goals, I will implement a thorough data science approach to understand customer turnover and create successful retention tactics. Figure 1.1 shows a full data science life cycle workflow that consists of six steps and follows a comprehensive approach, guaranteeing a systematic and data-driven analysis of the issue.

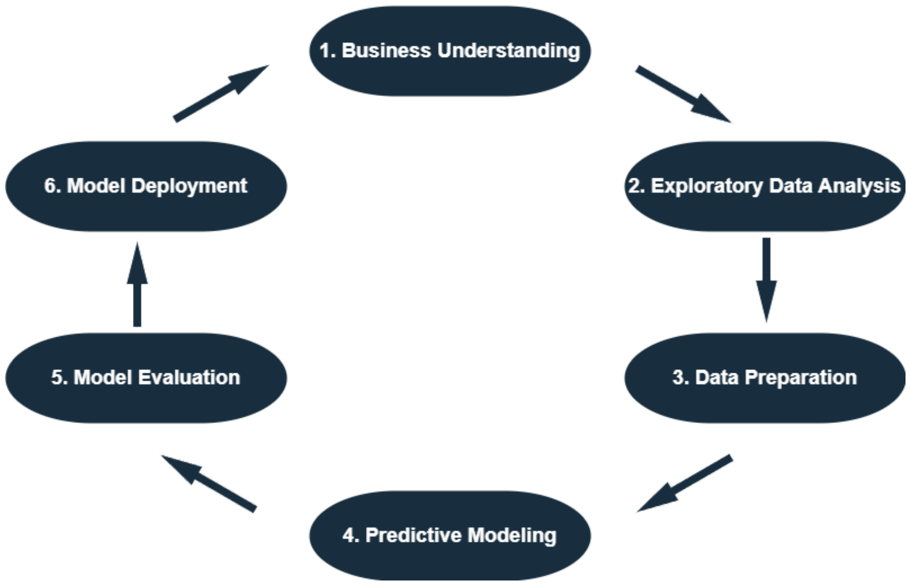

Figure 1.1: Data Science Life Cycle.

1. Business Understanding: The first phase will concentrate on clarifying the corporate objectives and matching them with the analytical aims. The main point is to comprehend the financial and operational consequences of client turnover and establish a definitive standard for effective retention efforts.

2. Exploratory Data Analysis: Through exploratory data analysis (EDA), it aims to discover the underlying patterns, relationships, and conclusions found from the telecom customer churn dataset, which can help guide predictive modeling and further strategic direction. The dataset comprises information on 7043 customers, encoded across 38 features that span demographic details, service usage, billing information,

and churn specifics. This comprehensive analysis forms the foundation for identifying key factors influencing customer behavior and retention in the telecom sector.

3. Data Preparation: In this step, raw data will be processed for all necessary tasks to generate the final dataset for machine learning model building and validation. Processing tasks include data cleaning, feature selections, data transformation, and data splitting into training and test sets.

4. Predictive Model Building: Insights from the EDA step will be used to develop and refine predictive models for forecasting customer attrition. All the popular machine learning models will be implemented and evaluated. The top 3 models will be selected and fine-tuned to improve their capacity to forecast outcomes, focusing on being easily understood and providing practical insights.

5. Model Evaluation: Models will be thoroughly evaluated using suitable performance metrics, namely accuracy, precision, recall, and F1 score. The goal is to choose a model that properly forecasts customer attrition and also offers insights into the probability of churn, allowing the organization to take proactive actions.

6. Model Deployment: Once the final model is validated and evaluated for robustness, it will be implemented in a production environment. This would enable real-time prediction of customer churn and the automation of efforts to retain customers, integrating data-driven decision-making into the company.

Chapter 2

Background

This chapter provides a summary of the literature review about the application of machine learning in the area of churn prediction. In addition, a brief description of the machine learning algorithms and techniques is also included.

2.1 Machine Learning in Churn Prediction

Churn prediction study has gained popularity in recent years due to its impact on corporations and businesses. The following sections summarize the recently published papers in this field of study.

A case study by Qureshi et al. (2013) on churn prediction in the mobile communication market, where people change from one company to another company. It describes the use of regression analysis, decision trees, artificial neural networks, and logistic regression to predict potential churners, looking into the historical data to get an idea of the pattern. The study used a dataset from the Customer DNA website, where usage data for 106,000 customers was provided over three months, along with total usage by the customers. Dealing with the problem of class imbalance in the dataset, the authors check how the different machine learning algorithms performed regarding real usage by the customer, while demonstrating that the decision trees were the best classifiers for identifying potential churners.

Sisodia et al. (2017) built a model for predicting employee churn rate based on HR analytics dataset, using five different machine learning algorithms, namely, linear support vector machine, decision tree classifier, random forest, k-nearest neighbor, and naïve bayes classifier. The authors evaluated the correlation between attributes, generated a histogram to contrast left employees with various factors, and proposed strategies to optimize employee attrition in organizations.

Ahmad et al. (2019) developed a churn prediction model using machine learning techniques on a big data platform to assist telecom operators in predicting customers who are likely to churn. The model incorporates features from social network analysis and achieves an AUC value of 93.3%. The dataset used for training and testing the model is obtained from SyriaTel telecom company and includes customer information over a period of 9 months. The model experiments with four algorithms, with the XGBoost algorithm yielding the best results for classification in churn prediction.

Ullah et al. (2019) proposed a churn prediction model using classification and clustering techniques to identify churn customers and determine the factors behind customer churn in the telecom sector. The model utilizes the Random Forest algorithm for churn classification and cosine similarity for grouping churn customers. It also employs feature selection techniques and attribute-selected classifier algorithm to identify significant churn factors. The evaluation of the proposed model shows improved churn classification and customer profiling. The results obtained from the model can help improve customer retention strategies, recommend relevant promotions, and enhance marketing campaigns.

Rahman and Kumar (2020)'s paper focuses on predicting customer churn in a commercial bank using efficient data mining methods. It discusses data transformation techniques and the use of various classification techniques such as k-nearest neighbor, support vector machine, decision tree, and random forest. The paper results show that oversampling improves the accuracy of decision tree and random forest classifiers, while support vector machine is not suitable for large amounts of data. The study also analyzes customer behavior to explore the likelihood of churn and compares the performance of different models, finding that the Random Forest model after oversampling achieves higher accuracy.

Lalwani et al. (2022) predicted customer churn prediction in the telecom industry by using machine learning techniques. The authors applied various predictive models such as logistic regression, naive bayes, support vector machine, random forest, decision trees, boosting, and ensemble techniques. The paper also discusses the use of K-fold cross-validation for hyperparameter tuning and preventing overfitting. The results show that Adaboost and XGBoost classifiers achieve the highest accuracy of 81.71% and 80.8%, respectively.

Geiler et al. (2022) focused on churn prediction in businesses and explored the performance of various supervised and semi-supervised learning methods and sampling approaches on publicly available datasets. The study suggests an ensemble approach should be used for churn prediction.

There are numerous studies on churn prediction using machine learning that have been published, with a predominant focus on the theoretical aspects of the field, such as algorithm comparison and enhancement. There is, however, a notable gap in the literature regarding the practical application of machine learning techniques in this churn prediction area. This research aims to bridge this gap by providing a comprehensive analysis of the practical implementation of machine learning in churn prediction, thereby contributing to the advancement of the field in a more applied manner.

2.2 Machine Learning Algorithms

2.2.1 Logistic Classification

Logistic regression is the simplest machine learning algorithm for binary as well as multiclass classification. It estimates the probability that an instance belongs to one of the classes as a function of input features using the logistic function (Eq. 2.1).

$$P(y = 1|\mathbf{x}) = \frac{e^{\beta_0+\beta_1 x_1+\beta_2 x_2+\ldots+\beta_k x_k}}{1 + e^{\beta_0+\beta_1 x_1+\beta_2 x_2+\ldots+\beta_k x_k}} \tag{2.1}$$

where:

- $P(y = 1|\mathbf{x})$ is the probability that the outcome y is 1 given the input features $\mathbf{x}$.
- $\mathbf{x} = (x_1, x_2, \ldots, x_k)$ is the vector of input features data.
- $\beta_0, \beta_1, \beta_2, \ldots, \beta_k$ are the model parameters.
- e is the base of the natural logarithm.

During training, the model parameters are optimized using techniques such as maximum likelihood estimation and gradient descent (Witten and James, 2013, Géron, 2022). It is very efficient to compute and highly interpretable.

2.2.2 The K-Nearest Neighbors (KNN)

KNN is a machine learning method used for both regression and classification tasks. KNN utilizes distance measurements to predict the most probable value of the target feature. The Euclidean distance is often used in KNN and it is calculated using Equation 2.2 (Witten and James, 2013). The predicted class in categorization is the most popular class among the k neighbors. The optimal k nearest neighbors are identified for the given case based on the cross-validation. This method is non-parametric. It is a categorization method based solely on examples, utilizing existing data without generalization. It is called a lazy learning algorithm since its stages and operations are executed during the query. The algorithm does not require any preparation. It is effective with low-dimensional data but less so with high-dimensional data. To use KNN for high-dimensional data, we may use principle component analysis before implementing KNN.

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{2.2}$$

where:

- $d(\mathbf{x}, \mathbf{y})$ is the Euclidean distance between points $\mathbf{x}$ and $\mathbf{y}$.
- $\sum_{i=1}^{n}$ denotes the summation over all dimensions from 1 to n.
- x_i and y_i are the coordinates of points $\mathbf{x}$ and $\mathbf{y}$ in the i-th dimension, respectively.
- $(x_i - y_i)^2$ is the squared difference between the i-th coordinates of the two points.

2.2.3 Support Vector Machines (SVM)

SVM is a supervised machine learning technique that has been widely used for classification purposes. It searches the hyperplane that accurately divides the majority of the training data into two classes, while it may incorrectly categorize a small number of observations. This is the optimal solution to the following optimization problem (Witten and James, 2013).

$$\underset{\beta_0,\beta_1,\ldots,\beta_p,\epsilon_1,\ldots,\epsilon_N,M}{\text{maximize}} M \tag{2.3}$$

$$\text{subject to} \quad \sum_{j=1}^{p} \beta_j^2 = 1 \tag{2.4}$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip}) \geq M(1 - \xi_i) \tag{2.5}$$

$$\xi_i \geq 0, \quad \sum_{i=1}^{n} \xi_i \leq C \tag{2.6}$$

where:

- M is the width of the margin.
- $\beta_0, \beta_1, \ldots, \beta_p$ are the coefficients of the hyperplane
- $\epsilon_1, \ldots, \epsilon_N$ individual observations that allow fall on the incorrect side of the margin or the hyperplane.
- $x_1, \ldots, x_n$ set of n training observations
- $y_1, \ldots, y_n$ associated class labels
- C is a nonnegative tuning parameter.

SVM can perform both linear and nonlinear classification. It can find a nonlinear separator, also known as a functional, by transforming the data into a higher-dimensional space. The algorithm then uses linear classification by selecting the support vectors to define the decision boundary or hyperplane. The SVM algorithm can utilize kernel functions, such as linear, polynomial, and radial basis function (RBF) to handle both linearly and nonlinearly separable data (Géron, 2022).

2.2.4 Naive Bayes

Naive Bayes is a type of probabilistic classifier model, which is based on the Bayes theorem. The Naive part comes from the assumption that observation characteristics are conditionally independent given the class label. It is stated mathematically as Equation 2.7 (Witten and James, 2013). Multiple class predictions are made feasible by probabilistic classifiers. Based on conditional probability, the decision is made.

$$Pr(Y = k \mid X = \mathbf{x}) = \frac{\pi_k \times f_{k1}(x_1) \times f_{k2}(x_2) \times \cdots \times f_{kp}(x_p)}{\sum_{l=1}^{K} \pi_l \times f_{l1}(x_1) \times f_{l2}(x_2) \times \cdots \times f_{lp}(x_p)} \tag{2.7}$$

for k = 1,...,K

where:

- Pr($Y = k \mid X = \mathbf{x}$) is posterior probability that an observation X belongs to k class.
- π_k is prior probability that a randomly chosen observation comes from the k class.
- f_{kj} is the density function of the jth predictor among observations in the kth class.

2.2.5 Decision Trees

Decision trees are powerful and versatile techniques used for classification and regression. It is commonly referred to as the Classification and Regression Trees (CART) method. The method is capable of handling data with large dimensionality and can process both numerical and categorical data. It operates by recursively partitioning the dataset into subsets based on the most informative features (Witten and James, 2013). Each internal node of the tree represents a decision point where a feature is evaluated, and each leaf node corresponds to a class label or a regression value. Decision trees are constructed using various criteria such as Gini impurity or information gain, to optimize the splitting process (Géron, 2022). They are interpretable models that facilitate human understanding of decision-making processes in complex datasets.

$$G_i = 1 - \sum_{k=1}^{n} p_{i,k}^2 \tag{2.8}$$

where:

- G_i is gini score of node i-th
- $p_{i,k}$ is the training instance ratio of class k in the i-th node.

2.2.6 Ensemble Classifiers

Ensemble method can help to minimize the variance and bias of separate models or classifiers by merging their predictions, resulting in more accurate and reliable models (Latha and Jeeva, 2019). There are three commonly used ensemble learning methods, namely bagging, boosting, and stacking, which can be utilized to enhance the machine learning process. In this section, we will delve into the details of each method, including its working nature, characteristics regarding data generation, training of baseline classifiers, and suitable fusion methods. We will also cover the advantages, disadvantages, and implementation challenges associated with each method.

Bagging

The bagging method, also referred to as bootstrap aggregating, is a data-driven algorithm that involves creating multiple subsets of data from the original dataset (Breiman, 1996). Bagging aims to generate diverse predictive models by adjusting the distribution of training datasets, where even small changes in the training data set can lead to significant changes in the model predictions. Bagging reduces variance, eliminates overfitting, and performs well on high-dimensional data. However, it also has some drawbacks such as being computationally expensive, having high bias, and reducing the interpretability of models (Bühlmann and Yu, 2002). The Random Forests algorithm is a notable example of the bagging technique (Breiman, 2001). Implementing the bagging method presents several challenges, such as determining the optimal number of base learners and subsets, the maximum number of bootstrap samples per subset, and the fusion method for integrating the outputs of the base classifiers using various voting methods. In summary, the bagging method employs parallel ensemble techniques with no data dependency, and the fusion methods depend on different voting methods to generate predictions. This approach generates B different bootstrapped training data sets. The algorithm is then trained on the b_{th} bootstrapped training set to predict a point x. The following equation is the bagging function:

$$f(\mathbf{x}) = \frac{1}{B} \sum_{b=1}^{B} f_b(\mathbf{x}) \tag{2.9}$$

where $f_b(\mathbf{x})$ is a weak learner

Boosting

The boosting approach is a sequential procedure in which each succeeding model attempts to correct the prior model's errors (Freund, Schapire, et al., 1996). Boosting employs multiple weak learners in a highly adaptive way, where each model in the sequence is fitted while giving more importance to observations that the previous models in the sequence handled poorly. Boosting can be used for both regression and classification problems and includes algorithms such as Adaptive Boosting (AdaBoost), Stochastic Gradient Boosting (SGB), and Extreme Gradient Boosting (XGB) (Freund et al., 2003, J. H. Friedman, 2001,and J. Friedman et al., 2000). Several studies have utilized various types of boosting, including AdaBoost for noise detection and speech feature extraction, and XGB for fake news classification (Sun et al., 2016, Asbai and Amrouche, 2017, and Haumahu et al., 2021). Boosting provides interpretability and helps reduce variance and bias in machine learning ensembles. However, the drawback of boosting is that each classifier must correct errors in the predecessors, and scaling sequential training can be challenging. Additionally, boosting is computationally costly, vulnerable to overfitting, and slower to train than bagging. To summarize, boosting is an ensemble learning technique that uses a sequential approach, where multiple learners learn sequentially with data dependency, and fusion methods rely on various voting methods. The boosting function is shown as follows:

$$f(x) = \sum_i \lambda_i g_i(x) \tag{2.10}$$

Where several classifiers $g_i(x)$ create a strong classifier $f(x)$. The inclusion of the shrinkage parameter λ in the model slows down the process, which enables a wider range of differently shaped trees to be utilized to address the residuals.

Stacking

The stacking method is a powerful model ensembling technique that combines multiple predictive models to generate a new model, also known as a meta-model (Džeroski and Ženko, 2004). The stacking model architecture consists of two or more base models (level 0 models) and a meta-model that integrates the predictions of the base models (level 1 models). The base models are fit on the training data and their predictions are compiled, while the meta-model learns how to best combine these predictions. One of the key benefits of stacking is its ability to provide a deeper understanding of the data, leading to increased precision and effectiveness in predictions. However, a major challenge with stacking is overfitting, which can occur when there are many predictors that all predict the same target. Additionally, multi-level stacking can be both data and time-intensive, as each layer adds multiple models. As the amount of available data grows exponentially, computation time complexity becomes an issue, and highly complex models may take months to run (Xiong et al., 2021). Another challenge with stacking is interpreting the final model, as well as identifying the appropriate number and baseline models that can be relied upon to generate better predictions from the dataset when designing a stacking ensemble from scratch. The problem of multi-label classification also poses issues such as overfitting and the curse of dimensionality due to the high dimensionality of the data (Elnagar et al., 2020). To sum up, stacking is a parallel ensemble method that creates baseline learners simultaneously without any data dependency. The meta-learning method determines the fusion techniques. Although stacking can be very successful, it poses several challenges such as overfitting, complexity, and interpretability, which must be taken into account when using it. The stacking model is as follows:

$$f_s(x) = \sum_{i=1}^{n} a_i f_i(x) \tag{2.11}$$

Stacking makes predictions from several models $(f_1, f_2, \ldots, f_n)$ to build a new model, where the new model is used to make predictions on the test dataset. Stacking aims to improve a model's predictive ability. To put it simply, stacking involves combining the predictions of multiple models by assigning weights to each prediction and adding them together by a linear combination of weights a_i.

2.3 Imbalanced Datasets Handling

Several methods can be employed to handle imbalanced datasets in machine learning classification:

- Resampling Techniques.
  - Oversampling: Involves augmenting the number of cases in the minority class by either duplicating existing samples or producing synthetic samples (Mohammed et al., 2020).
  - Undersampling: Reduce the number of majority class instances by randomly removing samples.
- Synthetic Minority Over-sampling Technique (SMOTE): Create artificial minority

samples by adding new data points between corresponding minority instances using interpolation (Elreedy and Atiya, 2019).

The above techniques may help to lessen the impact of the unbalanced dataset. It also uses stratified K-fold cross-validation and correct metrics to fully capture the impact of the unbalanced dataset on the performance of machine learning algorithms.

2.4 Stratified K-Fold Cross-Validation

Since the dataset is highly skewed, stratified K-fold cross-validation is used to evaluate ML models. In the K-fold cross-validation, the original dataset has first been divided into K-folds. Each fold has an equal number of training cases and an equal number of test cases. For model training, the train model is built based on K-1 folds as a training set and tested with the K-th test fold.

This process has been repeated K times. Each fold takes a turn to be the test set. Finally, the K-time modeled performance has been summarized. The major disadvantage with the K-fold cross-validation design is that when there is a severe class imbalance between classes in the original data set, the size of one class in the training fold is much less than the number of cases in the other class of the original data set (Pedregosa et al., 2011). The consequence of that is that the performance outcome in the validation evaluation is too optimistic and in favor of the model.

Therefore, stratified K-fold cross-validation is recommended in this case, and each fold should be stratified in the same way as the original data set. This can ensure that the number of each class in every fold is the same as the class distribution of the original data set. Figure 2.1 Stratified 5-fold cross-validation (Adapted from Pedregosa et al., 2011)

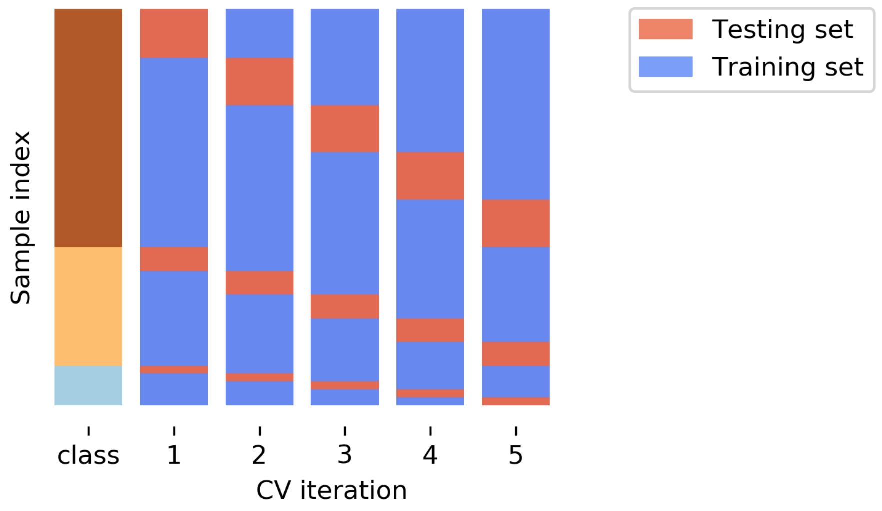

Figure 2.1: Stratified 5-Fold Cross Validation.

2.5 Model Performance Metrics

The performance of the models developed in this study is assessed using the confusion matrix and its derived metrics, namely accuracy, precision, recall, specificity, and F1 score. Figure 2.2, below, demonstrates the confusion matrix represented with actual and predicted outcome categories plotted against its axis (Adapted from Bryan, 2020).

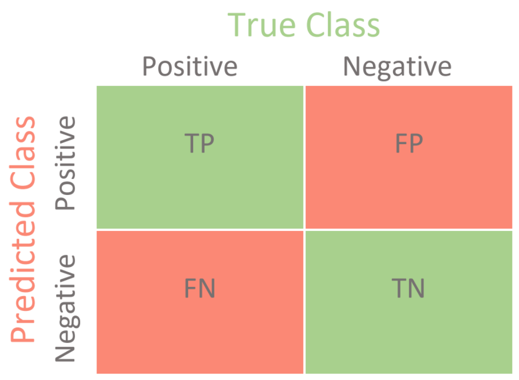

Figure 2.2: Confusion matrix.

2.5.1 Accuracy

Equation 2.12 calculates accuracy as the proportion of correctly predicted classes over the total number of instances in the dataset ((Witten & James, 2013)). Accuracy is used to measure the overall performance of a model but can be misleading in imbalanced datasets

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{2.12}$$

2.5.2 Precision

Precision shows the proportion of positive predictions that are truly positive (Witten & James, 2013). It is basically a ratio of correctly positively labeled to all positively labeled and can be calculated as follows:

$$Precision = \frac{TP}{TP + FP} \tag{2.13}$$

2.5.3 Recall

Recall has other names Sensitivity or True Positive Rate. It is a measure of the percentage of actual positive classes that are predicted as positive (Witten & James, 2013). It can be calculated as follows:

$$Recall = \frac{TP}{TP + FN} \tag{2.14}$$

2.5.4 F1 Score

The F1 Score is a measure that is calculated from precision and recall (Equation 2.15). It is often a preferred metric over accuracy when data is unbalanced.

$$\mathit{F1\ Score} = \frac{2 * Precision * Recall}{Precision + Recall} \tag{2.15}$$

Chapter 3

Data

This study plans to use several public churn datasets to compare and evaluate the performance of various machine learning algorithms.

The first data that I have done preliminary analysis is the telecom customer churn. This data can be accessed on the `Maven Analytics` website platform. This dataset comprises two tables, presented in CSV format. The Customer Churn table has data from a total of 7,043 customers who are associated with a telecommunications firm located in the state of California during the second quarter of the year 2022. Each entry in the dataset corresponds to an individual customer and has information about their demographic characteristics, geographical location, length of tenure, subscription services availed, and quarterly status (i.e., whether they joined, remained, or churned), among other relevant factors. The Zip Code Population table provides further data on the estimated populations of the California zip codes mentioned in the Customer Churn table.

The second dataset `internet service churn` is used for the analysis. This is a comprehensive aggregation of customer data from an internet service provider, encompassing 72,274 records with 11 unique attributes. These attributes span from binary indicators of service subscriptions, such as TV and movie packages, to continuous variables like subscription tenure (subscription-age), average billing amount (bill-avg), and service quality metrics (service-failure-count, download-avg, and upload-avg). There is a notable presence of missing data in key attributes such as remaining contract and internet speed measures (download-avg and upload-avg), which necessitates data cleaning or imputation for further analysis. At the core of the dataset is the churn attribute, marking the continuation or cessation of services by customers. The balanced nature of this variable provides a rich foundation for predictive modeling to discern the determinants of customer attrition, offering critical insights that can drive strategies to bolster customer retention and loyalty to the service provider.

The other dataset named bank customer churn records which accessible on Kaggle. This dataset comprises 10,000 records detailing an array of characteristics of bank clients. It spans demographic, financial, and service usage information with 18 attributes including surname, geography, gender, credit score, account balance, and number of banking products utilized. Other key aspects of the dataset involve customer satisfaction and engagement levels, represented by Satisfaction Score, Complain, and Exited statuses, alongside the Card Type and Point Earned fields that potentially reflect the effectiveness of customer loyalty programs. The dataset is complete with no missing entries, ensuring a straightforward preprocessing phase. With geography and card type reflecting categorical variables and others like credit score, age, balance, estimated salary, and points earned as quantitative data, the dataset is poised for diverse analytical approaches. The target variable, Exited, is binary and will facilitate the development of predictive churn models, allowing for the investigation into what drives customer attrition.

The dataset credit card churn is also used for this study and can be found on Kaggle. This dataset contains 10,127 entries and 23 attributes. It provides a rich source of information about credit card customers, encompassing personal demographics such as age, gender, and education level, as well as financial behaviors like credit limit usage and total transaction amount. Moreover, it tracks customer status changes with features like Attrition Flag, which indicates if a customer has left the bank. The presence of both categorical variables (e.g., Marital Status, Income Category) and continuous variables (e.g., Customer Age, Credit Limit) offers a multifaceted view of customer profiles. The data encapsulates not just the static attributes of customers but also their transactional behavior, providing insights into the dynamics of customer-bank interactions over time.

The other dataset used in this project is the `insurance churn prediction` datase. Which is a structured compilation of 33,908 instances with 17 attributes, each named generically from $_0$ to $_1$5, plus a label column. The attributes consist of a mix of continuous and categorical variables, with continuous features exhibiting a standard deviation close to one, indicating that they might have been standardized. The categorical variables have a limited number of unique values, pointing towards a structured categorical system within the dataset. There are no missing values, which implies that the data is clean and potentially ready for use in machine learning models without the need for initial data imputation. The labels are binary, suggesting that the dataset is likely prepared for a binary classification task. The distribution of values across different features varies, with some displaying a wide range of numbers and others showing a more narrow spread, indicative of diverse data types and scales represented within the dataset.

The dataset e-commerce churn is also used for this research and can be found on `Kaggle`. This dataset contains 5,629 entries and 20 columns, representing various attributes of customers in an e-commerce setting. The columns appear to be unnamed but contain a mix of numerical and categorical data, with features such as customer ID, gender, product categories, and payment methods. There are some missing values in several columns, indicating that data cleaning may be necessary before analysis. The dataset includes diverse features, such as the number of products purchased, customer demographics, and transaction details, which could be used to explore customer behavior and predict churn in an e-commerce context. The presence of a wide range of unique values in the product category and transaction columns suggests that the dataset captures a broad spectrum of customer interactions and preferences.

The customer churn in the real estate also included in this study. The dataset can be found on `Kaggle`. This dataset comprises 100,000 entries and 9 columns, capturing various attributes of customers in a real estate context. The attributes include CustomerID, Name, Age, Gender, Location, Subscription Length in Months, Monthly Bill, Total Usage in GB, and a Churn indicator. The dataset is well-balanced regarding the churn rate, with a mean value of approximately 0.5. It contains a diverse range of ages, subscription lengths, monthly bills, and usage amounts. The data is clean, with no missing values, and is likely ready for analysis or modeling tasks. With a mix of categorical and numerical attributes, this dataset could be used to explore factors influencing customer churn in the real estate sector and to develop predictive churn models.

Chapter 4

Methodology

4.1 Exploratory Data Analysis

This study uses a public dataset of telecom customer churn that can be accessed on the Maven Analytics website platform. The customer churn table has data from a total of 7,043 customers who are associated with a telecommunications firm located in the state of California during the second quarter of the year 2022. Each entry in the dataset corresponds to an individual customer and has information about their demographic characteristics, geographical location, length of tenure, subscription services availed, and quarterly status (i.e., whether they joined, remained, or churned), among other relevant factors.

This section presents a detailed exploratory data analysis (EDA) of a telecom customer churn dataset, aiming to uncover underlying patterns, relationships, and insights that could inform subsequent predictive modeling and strategic decision-making processes. This comprehensive analysis forms the foundation for identifying key factors influencing customer behavior and retention in the telecom sector.

The dataset comprises both categorical and numerical attributes. It describes in detail customer demographics (e.g., gender, age), service (e.g., Internet service, contract type), financial transactions (e.g., monthly charge, total charges), and churn information (e.g., customer status, churn reason). Initial inspection revealed the presence of missing values predominantly in the churn-related columns, with 5174 missing values in both the churn reason and churn category columns. However, the missing values are loyal customers. There are also cases of missing 1526 values in columns that are related to customers who are not subscribed to the Internet service. The last missing data is observed in the phone service-related columns, with 682 missing values. Since these missing data are a large proportion of datasets, they will be properly handled in the data processing step.

Table 4.1: Descriptive Statistics of Key Numerical Variables

	Age	**Tenure in Months**	**Total Charges**	**Total Revenue**
count	7043.00	7043.00	7043.00	7043.00
mean	46.51	32.39	2280.38	3034.38
std	16.75	24.54	2266.22	2865.20
min	19.00	1.00	18.80	21.36
25%	32.00	9.00	400.15	605.61
50%	46.00	29.00	1394.55	2108.64
75%	60.00	55.00	3786.60	4801.15
max	80.00	72.00	8684.80	11979.34

It is worth noting that descriptive statistics of numeric variables showed various customer demographics as well as service interactions. Table 4.1 is a summary of statistics for key numeric variables. Data gives us an idea that there was a broad spectrum of telecom services for customers of various age groups between 19 and 80 years old, with an average age of around 46 years. Tenure varied from 1 to 72 months, which exhibits the fact that customers were new and long-term ones spread across our variables. Similar to tenure, total charges for services varied between 18.8 and 8684.8 dollars, with a mean of around 2280.38 dollars. The pattern was similar in the total revenue, having minimum, maximum, and average values of 21.36, 11979.34, and 3034.38 dollars, respectively.

The EDA of customer churn status, churn category, and churn reason were central to this analysis. That helps to understand how churn occurs and which fields need more attention to help retain more customers. There are 3 different categories of customer status in this dataset, namely, stayed, churned, and joined (Figure 4.1). Customers stayed with the service are much more than those who churned or joined. This shows the current customer base is in good numbers. A considerable count has also been shown under the churned category, which shows an area of concern. This is where customer retention strategies can be used to keep customers using the service. The joined category count is much lower compared to the other two categories. This shows the rate of new customers joining the service is relatively lower than the rate of customers who have been using it for a long period already, or maybe the number of customers who stopped using the service.

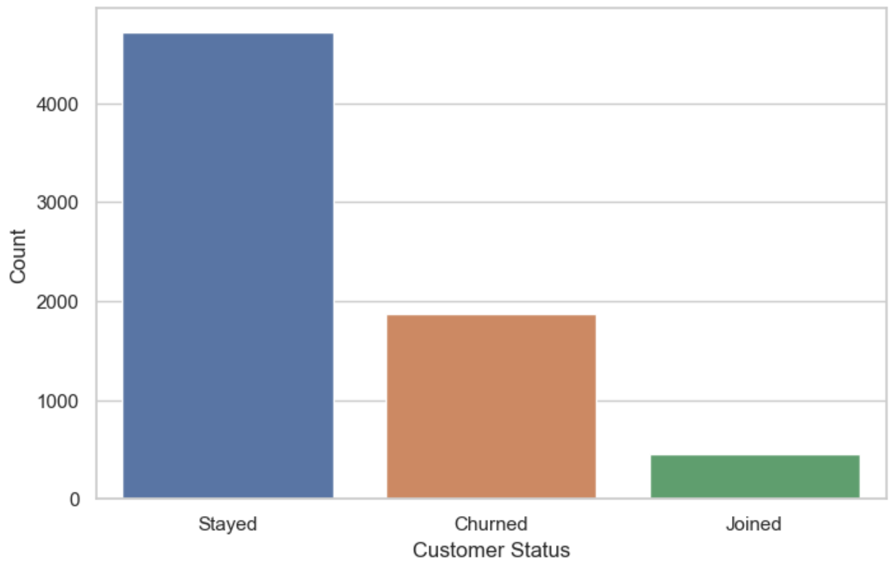

Figure 4.1: Customer Status.

Figure 4.2 shows reasons for customer churn in five categories: competitor, dissatisfaction, attitude, price, and others, with the competitor category representing the highest count. The competitive entities have likely been able to insist that customers move on to them instead of using the presented service. Arguably, dissatisfaction is the second notable reason for churn, while attitude and price were also escape options, although less used than the other two. Besides, Other leaves some unspecified reasons, which may or may not implicitly indicate customer dissatisfaction.

By looking at the reason for customer churn, there is a large number of customers left because of attractive product offerings and better deals available in the market. Other primary reasons for customer churn include offers with better service delivery, such as better speed rate, and soft factors such as the attitude of the dealer or technical support person. Prices are high, and product dissatisfaction ranks third with a high number of outgoing deals. This breakdown of details helps corporations plan precise interventions in their policies. Overall, these analyses enable the business to act quickly to retain customers. They address where it should direct its customer satisfaction and retention efforts: in competitive offers, service attitudes, and pricing strategies.

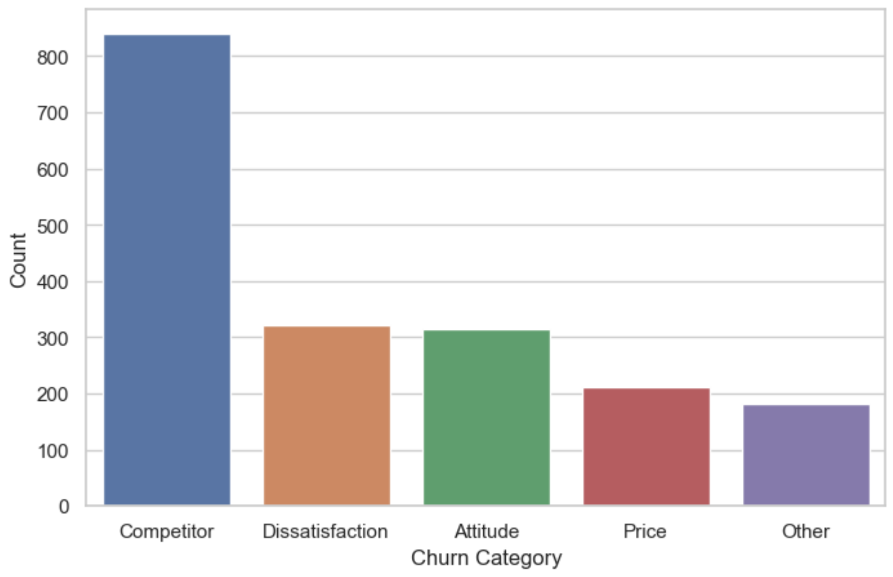

Figure 4.2: Customer Churn Categories.

The heat map (Figure 4.3) shows the correlation coefficients among numerical variables in the telecommunications data set. It can be seen that there is a strong correlation between total revenue and total charges, with a correlation of 0.97. This means that total charges is a key component of total revenue. It is noted that total revenue and total long-distance charges also have a strong correlation (0.78), showing that total long-distance charges is a major component of total revenue. There is a strong positive correlation (0.83) between tenure in months and total charges. It shows how the longer a customer stays with a company, the higher his or her total charges become. In addition, the average monthly GB download has a moderate negative correlation with age (-0.57), which could be an interesting aspect to explore in understanding customer behavior. To sum up, with the high correlations between the above variables, the use of these features together in predictive modeling could be subject to multicollinearity. Therefore, this potential issue should be addressed properly in the data processing step.

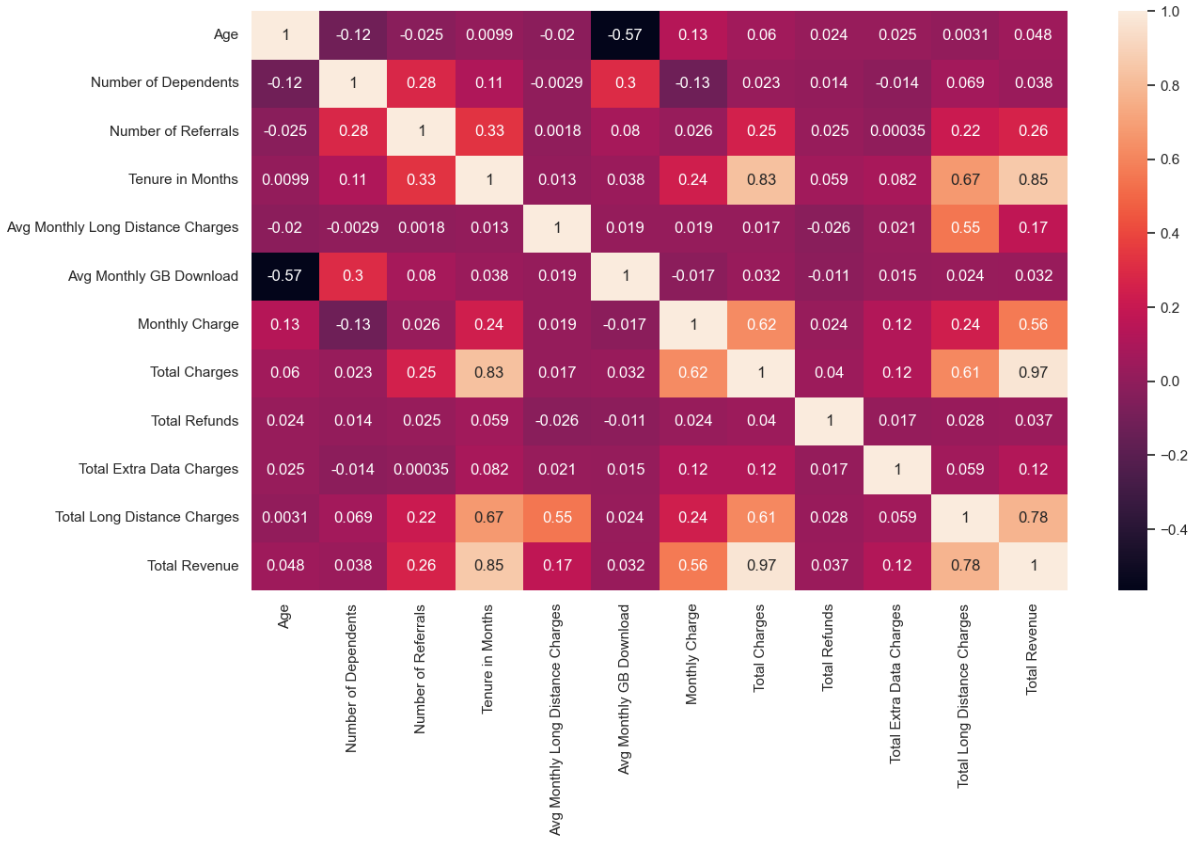

Figure 4.3: Correlations Among Numerical Features.

Data visualization on categorical features like the type of contract, customer's gender, and marital status provides information on the composition and preferences of the customers. There is a large percentage of customers who only signed a month-to-month contract (Figure 4.4). This group also has the highest churn rate among the contract types. Whereas, the number of customers with long-term contracts is much lower; however, they are the most loyal customers. Data suggests that overall, the mix of male and female customers is similar. This shows that the popularity of the given services is not connected with gender in any real way. In addition, data reveals that unmarried customers' churn rate is higher than their married counterparts. One possible explanation for this is that perhaps married customers have more commitments in their lives and therefore have a lower chance of churning.

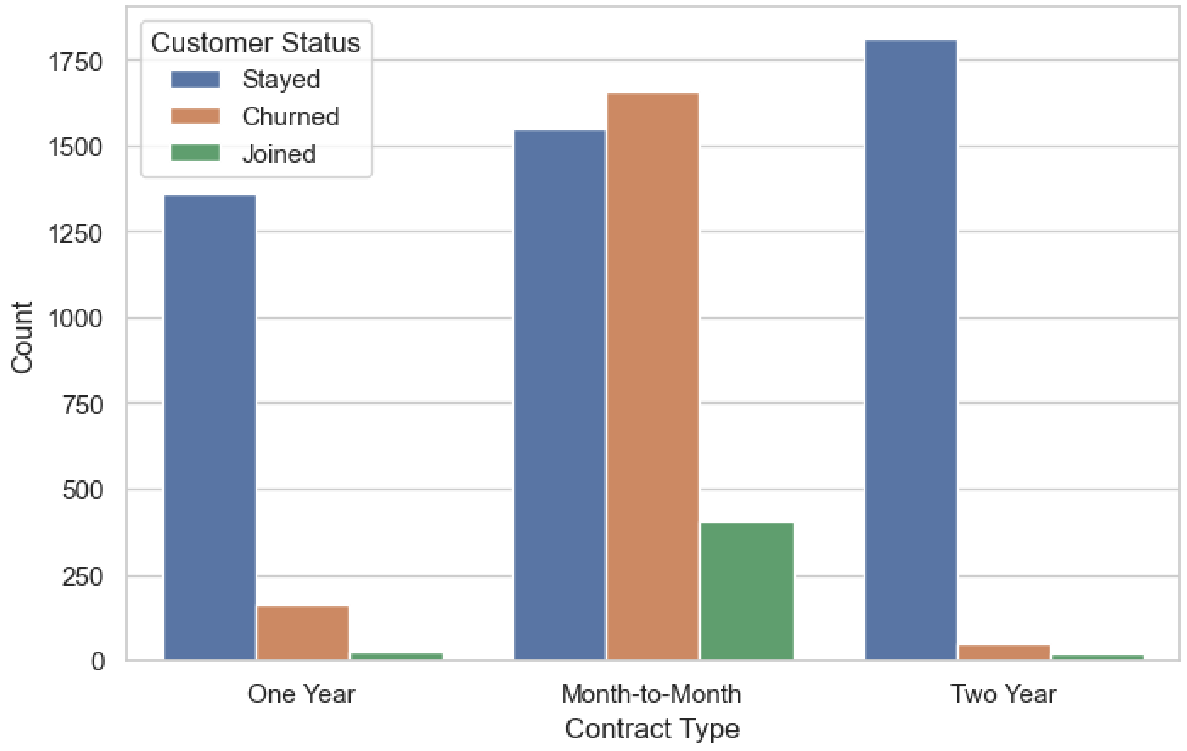

Figure 4.4: Customer Status by Contract Types.

Exploratory data analysis of the telecom customer churn dataset helps the company understand the customers in detail, like their demography, service usage, financial transactions, and patterns of churn. Such insights also help with further investigation and prediction of churn. Furthermore, this analysis helps develop a strategy for higher customer loyalty and retention. Ultimately, it is the insights formed from this analysis that will guide the managers and help them make better decisions in their business.

4.2 Data Preparation

Data preparation is the next step after data collection and before machine learning model building in this workflow. It includes missing value treatment, scaling numerical variables, encoding categorical features, imbalanced data handling, data splitting, etc. All these are vital steps for data preparation before training a machine learning algorithm.

4.2.1 Data Splitting

Splitting the dataset into train and test sets is done first in the data preparation to avoid any potential data leakage during this step. Data leakage occurs when training models can access information outside of the training dataset (Shabtai et al., 2012). This can lead to overestimating models' performance during the training process. Once the model is deployed in production, its performance can decrease significantly. The predictions turn out to be far less reliable than the performance of the model during training would suggest. Steps can be taken to eliminate what is called data leakage so that the performance during training is a better reflection of what the model can do with data it has not seen already.

Since the telecom customer churn dataset is unbalanced data, stratified splitting was applied to ensure that the class distribution of training and test sets is similar (Joseph and Vakayil, 2022). The initial dataset was split into 75 percent and 25 percent for the train and test sets, respectively. The random state is set to make sure the experiment result is reproducible.

4.2.2 Handling Missing Values

First, some non-informative columns, namely, customer id, city, zip code, latitude, and longitude were dropped, assuming that these variables simply do not offer predictive power. In addition, direct churn-related columns, churn category, and churn reason were also removed because they are directly related to churn and only available after customer churn. If these two columns were included in the model, it would introduce a very optimistic model for churn prediction.

The process continues with the imputation of missing values for categorical variables. As pointed out in the previous exploratory data analysis, for service-related features like multiple lines, internet type, and online security, among others, we will fill the missing entries with No because these customers do not have a service. By replacing missing categorical data with meaningful labels, we avoid introducing bias arising from dropping rows that contain missing data, which may lead to the loss of important information.

For a continuous variable such as average monthly long distance charges or average monthly GB download, the imputation method used is zero filling. The rationale is that these customers do not subscribe to phone or internet service.

4.2.3 Scaling Numerical Variables

The numerical columns were scaled using min-max scaling. In this study, the MinMaxScaler function from the sklearn.preprocessing module is used to implement the scaling task. It calculates the difference between each value and the minimum feature value and then divides it by the difference between the maximum and minimum feature values. This scaling process transformed all the numerical features into a range between 0 and 1, which would help many machine learning algorithms that are very sensitive to the scale of the input features (Alshdaifat et al., 2021).

4.2.4 Encoding Categorical Variables

In the encoding phase, binary categorical variables such as gender, married, phone service, and paperless billing are manually mapped to numeric binary values, with a straightforward mapping of female and yes to 1, and male and no to 0. This binary encoding transforms the categorical data into a format suitable for machine learning algorithms, which require numerical input.

The remaining categorical variables undergo one-hot encoding, a process that converts categorical variable values into a form that could be provided to machine learning algorithms to do a better job at prediction. The OneHotEncoder creates a binary column for each category and returns a sparse matrix or dense array (Pedregosa et al., 2011). By employing one-hot encoding, the model can interpret the data without falsely attributing order or priority where it does not exist (Seger, 2018).

4.2.5 Data Sampling

The SMOTE technique was implemented to see whether it could help minimize class imbalance problems in the churn dataset or not. SMOTE should apply only to the training set to prevent leaking information during the training process. Which is essential for the model's performance evaluation.

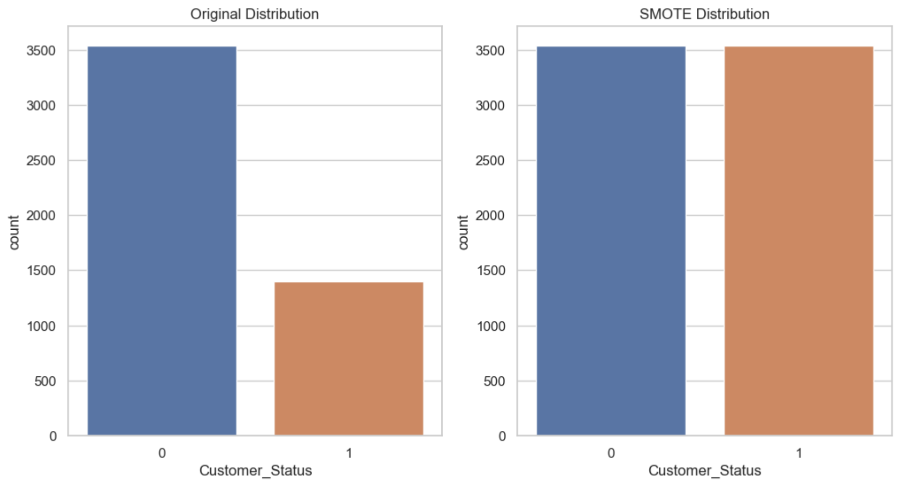

Figure 4.5: The training class samples before and after apply SMOTE.

Applying SMOTE to the training set can be done using a library such as imbalanced-learn in Python. The minority class is oversampled so that the training set contains synthetic samples that are nearest to the samples of the minority class. After training on the newly balanced training set, the model is assessed on the untouched testing set to test its ability to generalize to new data. Figure 4.5 shows the target class distribution with and without applying the SMOTE technique.

4.3 Predictive Modeling

In this study, a large number of machine learning models were implemented to find which model was best suited to the research tasks. Many of these models are ensembles that consist of other machine learning algorithms.

First, logistic regression is selected for the model benchmark. Then the support vector machine was also included. This model works well in high dimensions without scaling. For comparability purposes, KNN, gaussian naive bayes, decision trees, and random forest classifiers were selected. Among these algorithms, KNN is a simple model that also works effectively in classification. The Gaussian Naive Bayes classifier is a naturally probabilistic approach known for its overall good performance in probabilistic classification. The decision tree is an easy-to-interpret model, and the random forest is robust and deals with overfitting.

As an additional benchmark, several boosting algorithms were also implemented, including gradient boosting, AdaBoost, and bagging classifiers. These ensemble models are built from many weak learner models into strong ones. A multi-layer perceptron (MLP) classifier, which is an artificial neural network, was also employed. This model is capable of inferring highly non-linear relationships. Moreover, this study includes some highly sophisticated ensemble methods, such as XGBoost and LightGBM, that have proven to be very fast and highly competitive in many machine learning competitions.

Each of the models was initialized with a random seed to ensure that the reproduction of the results was reproducible. The model's performance is evaluated using multiple different metrics: accuracy, precision, recall, and F1 score, to give the best evaluation available. The final model will be decided upon by weighing all the performance measures as well as other considerations such as interpretability and computational efficiency.

4.4 Model Evaluation

4.4.1 Classifiers Performance Evaluation

All the models were evaluated by stratified k-fold cross-validation with 5 splits. This approach ensures that the number of samples for each class is about the same in each split. Such splitting can better evaluate the model performance than a single train-test split, especially for imbalanced data. The metrics of model evaluation are accuracy, precision, recall, and F1 score.

For each step of the cross-validation, the initial training data was split into training and validation sets. Then the model was trained on the training set and evaluated on the validation set, and the obtained score for each metric was recorded. This process was repeated for all training-validation splits, and the obtained set of scores was then averaged across folds to get a more stable estimate of the model's performance.

After cross-validation, the models were retrained on the entire initial training set and evaluated on the unseen test set to assess generalization performance. The models were then evaluated on the test set using the same metrics used in the previous evaluation.

The overall evaluation of the model helped us in comparing all types of machine learning algorithms and predicting customer churn on various parameters, giving us an upper hand in determining which model can be used in a real-life scenario if we want high accuracy without biases in our data. The results of both cross-validation and test set evaluation helped us arrive at the best model to suit our purpose.

4.4.2 Hyperparameter Tuning

This section conducts hyperparameter tuning for the top three classifiers, LightGBM, AdaBoost, and Gradient Boosting, using GridSearchCV and RandomisedSearchCV. The tuning was aimed at optimizing the hyperparameters to improve the model's performance.

Hyperparameters for the LightGBM classifier include boosting type, number of leaves, maximum depth, learning rate, number of estimators, and regularization parameters. While AdaBoost tunes the number of estimators and learning rate, Gradient Boosting tunes the loss function, learning rate, number of estimators, and subsample ratio.

GridSearchCV was used to do an exhaustive search over the parameter grid for each classifier. For each classifier, it evaluates all possible combinations of parameters and picks the best. RandomizedSearchCV used a fixed number of parameters randomly sampled. It is less intensive to compute than GridSearchCV, but sometimes it can find a good approximation of the best parameters.

Once the tuning was done, the best parameters for each classifier were determined and re-trained on the entire training dataset. Finally, the retrained models were assessed on the unseen test dataset, where the ability to generalize can be estimated. Performance can be evaluated using two metrics: accuracy and F1 score.

These provided the results of the hyperparameter tuning and model evaluation, giving insight into what was the best way to set each of the classifiers for the dataset and allowing us to maximize the predictive performance of the models, which in turn would make them more useful for real-world applications.

4.5 Model Deployment

4.5.1 Model Deployment Overview

At this point, we are ready to build the last machine learning model and deploy it to a production environment to enable it for inference. There are various ways to deploy models, each with a specific set of considerations and benefits. The following is an overview of the various deployment options:

Cloud Deployment: Amazon Web Services (AWS), Azure, and Google Cloud all have machine learning model deployment services where models can be deployed and made accessible to applications. They typically have scaling, security, and ease of integration with other cloud services (García et al., 2020).

On-Premises Deployment: The model runs on the organization's servers. The benefit of this approach is that the deployment environment and possibly the security of the data are more under the control of the organization itself. However, it also means that the organization itself needs to manage and maintain that infrastructure. It could also lack the scalability of other models that are running on the infrastructure of a cloud provider (Krishnan, 2013).

Edge deployment: Models can be deployed on edge devices, like Internet of Things devices or smartphones, where inference can be done without the need to have a constant connection with a central server. This makes low-latency responses possible even in situations where connectivity is lacking (Lai and Suda, 2018).

Model-as-a-Service (MaaS): The model is deployed as a service via APIs (application programming interfaces) either through custom web services or stored in containers like Docker. MaaS offers access to a model via APIs, and also flexibility to run in the cloud or on-premises (Ilić et al., 2024).

Embedded Deployment: The model is integrated directly into the application code, which might be the ideal fit for simple models or when the application has a tight integration requirement but is harder to update (Lin et al., 2018).

Batch Inference: Models can be deployed to perform batch inference on large datasets at scheduled intervals, such as weekly or monthly. This is how most data analytics and reporting applications function (Argesanu and Andreescu, 2021).

It is therefore a trade-off between how much coding time you are willing to invest and how well various deployment strategies will integrate. So, it all comes down to scalability, latency, security, and maintainability when choosing a deployment technique.

4.5.2 Model Deployment on Azure Cloud

After an ML model has been developed and tested, it will be deployed in a production setting. Which makes it accessible and usable so that it can make predictions. The following paragraphs describe deploying an ML model on the Azure Cloud.

Model Serialization: Serializing the trained model involves writing the trained model to disk, which is needed so that the model can be later loaded and used for prediction. This process currently uses Python libraries like Joblib or Pickle. The model serialized file is saved in Azure Blob Storage.

A Scoring Script: This is a script that loads the model from the byte stream and uses it to make predictions on the new data. It normally contains functions for data preprocessing, model loading, and prediction. To try it out, run this script locally.

Before deploying the model on Azure, it is essential to establish various resources within the Azure ecosystem. Firstly, an Azure Machine Learning Workspace is created to manage and organize the resources related to Azure Machine Learning. Subsequently, a compute target is designated, which can either be Azure Container Instances (ACI) for deployments with lighter usage or Azure Kubernetes Service (AKS) for deployments requiring scalability and a more production-grade environment. Lastly, an Azure Machine Learning Endpoint is configured to expose the deployed model as a REST API. This endpoint facilitates interaction with the model by enabling clients to submit data and receive predictions in return.

Deploy the Model: Using an inference configuration that specifies the scoring script and dependencies and a compute target, we deploy the model for a real-time endpoint either with ACI or AKS using the Azure ML SDK for Python.

Once the model deployment is successfully deployed, we should check whether the endpoint works as designed or not by passing a few sample input requests and checking whether the responses are as expected. The model still needs to be monitored to see if it is performing the desired function correctly after deployment. Azure ML gives tools for monitoring the health and usage of the model. The prediction function might also need to be periodically updated and maintained to keep it working.

After these steps have been completed, the machine learning model has been deployed on the Azure Cloud and can now be used to make predictions in a production environment.

Chapter 5

Results

5.1 Exploratory Data Analysis Findings

The analyses revealed several key insights from the dataset that are crucial for focusing marketing efforts toward understanding and building up customer loyalty.

The customer status is further categorized into three categories, namely, stayed, churned, and joined, with the majority of customers belonging to stayed having a significantly higher count compared to the other two. On the other hand, the category of churned sounds like a bell for customer retention; nonetheless, a considerable number of customers shifted from the current connection to another. The reasons for customer churn are further categorized into five groups: competitors, dissatisfaction, attitude, price, and others, implying that competitors having an attractive offer and better service delivery is the common cause of churn.

The correlation analysis showed a strong positive correlation between total revenue and total charges (0.97), total revenue and total long distance charges (0.78), and tenure in months and total charges (0.83). The moderately negative correlation between average monthly GB downloads and age (-0.57) indicates a lower number of downloads among the elderly.

The contract types analysis has indicated that the majority of customers are on month-to-month contracts and also tend to churn. In contrast, the long-term contract had more customer loyalty. Interestingly, unmarried customers tend to have a higher rate of churn compared to married customers.

The churn analysis revealed profound insights towards strategic planning to effectively build up customer loyalty, focusing on the areas of competitive offers, service quality, and pricing strategies. Additionally, the personalized campaigns towards customers on month-to-month contracts and younger demographics could have shown a significantly greater outcome.

5.2 Model Performance Evaluation

Scores of five-fold cross-validation on the training dataset are summarized in Table 5.1. The Gradient Boosting and LightGBM classifiers had the highest F1 scores of 0.76, demonstrating the best possible precision/recall balance. The same two classifiers showed an accuracy of 0.87, predicting the right label more often than other classifiers. In fourth and fifth place were the AdaBoost and XGBoost classifiers, with F1 scores of 0.74 and accuracy scores of 0.86. This is also indicative that ensemble methods, which create a better learner out of an ensemble of weak learners, are especially favored for this dataset.

Next, logistic regression and random forest achieved as good performance as the baseline, with a F1 score of 0.73 each. The MLP and Bagging classifiers reported an F1 score of 0.71 each. SVC received an F1 score of 0.70. GaussianNB, Decision Tree, and KNeighbors classifiers ranked a bit lower, with F1 scores of 0.69, 0.68, and 0.64, respectively.

Overall, the cross-validation scores suggest that ensemble methods are appropriate for this problem. Three methods score well on almost all metrics: accuracy, precision, recall, and F1 score, which are Gradient Boosting, LightGBM and AdaBoost.

Table 5.1: Cross validation Scores on Training Dataset

Classifiers	**Accuracy**	**Precision**	**Recall**	**F1 Score**
Gradient Boosting	0.87	0.82	0.71	0.76
LightGBM	0.87	0.80	0.71	0.76
AdaBoost	0.86	0.77	0.72	0.74
XGBoost	0.86	0.79	0.70	0.74
Logistic Regression	0.85	0.74	0.71	0.73
Random Forest	0.86	0.82	0.66	0.73
Bagging	0.85	0.80	0.63	0.71
MLP	0.84	0.73	0.69	0.71
SVC	0.84	0.74	0.66	0.70
GaussianNB	0.80	0.62	0.79	0.69
Decision Tree	0.82	0.67	0.68	0.68
KNeighbors	0.80	0.64	0.65	0.64

Each of the chosen classifiers performance was evaluated by fivefold cross-validation on the training set and thereafter evaluated for generalization on the unseen test set. The cross-validation scores in Table 5.2 served as a measure of their generalization capacity, and the test set scores in Table 5.3 as an indicator of their capacity for unseen data.

The Gradient Boosting and LightGBM classifiers showed good performance in both phases of evaluation, with top F1 scores among the classifiers. Gradient Boosting gave the F1 score of 0.76, which rose from 0.76 in the case of cross-validation to 0.79 on a test set, thus showing strong performances on unseen data. LightGBM, in turn, demonstrated stability, with an F1 score of 0.76 in both phases. AdaBoost Classifier is next best, with 0.74 on cross-validation and 0.76 on the test set; next after that is XGBoost: 0.74 on cross-validation and 0.76 on the test set.

Cross-validation on Logistic Regression and Random Forest classifiers revealed moderate performance, with F1 increasing to 0.76 for both. The Bagging Classifier, the MLP Classifier, and the SVC all show the same trends, though the exact values of F1 scores change when cross-validation is compared to the test set score. Even GaussianNB, the Decision Tree, and the KNeighbors Classifier, which are less effective than the other four classifiers in terms of their F1 scores, still reveal the overall character of the dataset.

Table 5.2: Test Set Scores from Models Trained on Training Dataset

Classifiers	**Accuracy**	**Precision**	**Recall**	**F1 Score**
Gradient Boosting	0.89	0.83	0.76	0.79
LightGBM	0.88	0.82	0.75	0.79
AdaBoost	0.87	0.78	0.76	0.77
Logistic Regression	0.87	0.76	0.76	0.76
Random Forest	0.87	0.82	0.70	0.76
XGBoost	0.87	0.79	0.74	0.76
SVC	0.87	0.79	0.72	0.75
Bagging	0.87	0.81	0.69	0.74
MLP	0.86	0.78	0.70	0.74
GaussianNB	0.82	0.64	0.84	0.72
Decision Tree	0.83	0.68	0.75	0.71
KNeighbors	0.79	0.63	0.64	0.63

Model Performance Evaluation with SMOTE

Table 5.3 illustrates the five-fold cross-validation scores of selected algorithms on a SMOTE-based training dataset. There was an improvement in the performance of many different classifiers. All classifiers, in particular Random Forest, XGBoost, and LightGBM, had an accuracy of 0.90. Furthermore, XGBoost and LightingBoost also reached a precision, recall, and F1 Score of 0.90. Comparing the two, we note that Gradient Boosting and AdaBoost's accuracy and precision are lower, and their F1 scores were 0.89 and 0.88, accordingly. The bagging classifier again had a perfect precision of 0.90. However, it had slightly lower recall and accuracy. The MLP classifier, SVC, and

Decision Tree also had very good performance with SMOTE. Accuracy was improved from 0.69 to 0.87. Logistic Regression and KNeighbors Classifier were relatively low performers, but they still showed some improvements with SMOTE. Particularly for the KNeighbors Classifier, the precision and F1 Score were at 0.76 and 0.84 when SMOTE was used (previously 0.64). Even more importantly, SMOTE changed the recall from 0.65 to 0.93.

Table 5.3: Cross-validation Scores on SMOTE Training Dataset:

Classifiers	**Accuracy**	**Precision**	**Recall**	**F1 Score**
Random Forest	0.90	0.89	0.91	0.90
XGBoost	0.90	0.90	0.90	0.90
LightGBM	0.90	0.90	0.90	0.90
Gradient Boosting	0.89	0.88	0.90	0.89
AdaBoost	0.88	0.86	0.89	0.88
Bagging	0.88	0.90	0.87	0.88
MLP	0.87	0.86	0.90	0.88
SVC	0.86	0.84	0.90	0.87
Decision Tree	0.85	0.85	0.87	0.86
Logistic Regression	0.84	0.82	0.88	0.85
KNeighbors	0.82	0.76	0.93	0.84
GaussianNB	0.82	0.82	0.83	0.82

Table 5.4 shows scores evaluated on the unseen test set of selected classifiers when they were trained on the SMOTE-based dataset. The five-fold cross-validation scores on the training set show that the models did better when they were trained on the training dataset with SMOTE, but the scores on the unseen test do not show a big improvement. Similar to other previous evaluations, the top three performers were also LightGBM, Gradient Boosting, and AdaBoost classifiers.

The LightGBM classifier had the highest accuracy, which is 0.88, and precision, recall, and F1's scores were 0.78, 0.80, and 0.79, respectively. Following it are Gradient

Table 5.4: Test Set Scores from Models Trained on SMOTE Training Dataset

Classifiers	**Accuracy**	**Precision**	**Recall**	**F1 Score**
LightGBM	0.88	0.78	0.80	0.79
Gradient Boosting	0.87	0.73	0.84	0.78
AdaBoost	0.86	0.71	0.86	0.78
XGBoost	0.87	0.77	0.79	0.78
Random Forest	0.87	0.76	0.78	0.77
Logistic Regression	0.83	0.65	0.88	0.75
SVC	0.84	0.68	0.83	0.74
Bagging	0.86	0.76	0.73	0.74
MLP Classifier	0.84	0.72	0.73	0.73
GaussianNB	0.82	0.64	0.81	0.72
Decision Tree	0.81	0.64	0.73	0.68
KNeighbors	0.74	0.52	0.81	0.64

Boosting and AdaBoost classifiers with an equal F1 score of 0.78 and a mildly lower accuracy. However, their precision and recall values were different. The XGBoost and Random Forest also gave a very good score, with an accuracy of 0.87, but these were almost equivalent to the F1 scores. The logistic regression and SVC were also very moderate, but the logistic regression showed a high recall score of 0.88. The Bagging Classifier, the MLP Classifier, and the GaussianNB gave almost equal results in terms of accuracy and F1 scores. However, the Decision Tree and KNeighbors Classifier gave the lowest scores, with the KNeighbors Classifier scoring a particularly low accuracy score of 0.74.

The three best classifiers, LightGBM, Gradient Boosting, and AdaBoost, were selected based on five-fold cross-validation results on the training dataset with and without SMOTE and, more importantly, results on test data that had not been seen before. These three classifiers will now have their hyperparameters tuned to make them even better at making predictions.

5.3 Hyperparameter Tuning Result

Tables 5.5, 5.6, and 5.7 provide a summary of the optimal parameters, which were derived from hyperparameter tuning for LightGBM, Gradient Boosting, and AdaBoost, respectively.

Table 5.5: Best Parameters for LightGBM Classifier

Parameter	**Value**
boosting_type	dart
class_weight	None
colsample_bytree	0.637
importance_type	gain
learning_rate	0.108
max_depth	4
min_child_samples	15
min_child_weight	0.1
min_split_gain	0.035
n_estimators	233
num_leaves	59
objective	binary
reg_alpha	0.817
reg_lambda	0.702
subsample	0.918
subsample_for_bin	222656
subsample_freq	1

The best parameters for the LightGBM model include a boosting type of dart, which indicates the use of dropouts that meet multiple additive regression trees. The model prefers a maximum depth of 4, allowing for moderate complexity in the decision trees, and a learning rate of approximately 0.108, balancing the trade-off between speed and accuracy. Notably, the model selects a relatively high num leaves value of 59, suggesting that it benefits from more granular decision rules in the trees.

Table 5.6: Best Parameters for Gradient Boosting Classifier

Parameter	**Value**
criterion	friedman_mse
learning_rate	0.154
loss	exponential
max_depth	8
min_impurity_decrease	0.036
min_samples_leaf	3
min_samples_split	5
min_weight_fraction_leaf	0.090
n_estimators	56
subsample	0.832

In the Gradient Boosting model, the best parameters include a loss function of exponential, which is commonly used for classification tasks. The model has a relatively high learning rate of 0.154, indicating faster convergence, and a maximum depth of 8, allowing for complex interactions between features. Additionally, the subsample parameter is set to 0.832, suggesting that the model uses a fraction of the data for fitting individual base learners to reduce variance.

Table 5.7: Best Parameters for AdaBoost Classifier

Parameter	**Value**
algorithm	SAMME.R
learning_rate	0.602
n_estimators	66

In the case of the AdaBoost model, the selected algorithm is SAMME.R, which is a real-valued version of the SAMME boosting algorithm. The learning rate is set to approximately 0.602, indicating a relatively aggressive weight update for each successive weak learner. The model utilizes 66 estimators, which represents the number of weak learners to be used in the boosting process.

Then these parameters were used to retrain the models and re-evaluate them. Table 5.8 shows the scores of three classifiers on the unseen test set. There was only a slight performance improvement in the LightGBM model when compared with the model run on default parameters. The results reveal that all three models performed well on this telecom churn dataset. However, LightGBM outperformed other classifiers. Therefore, it was selected to be deployed in the production environment.

Table 5.8: Performance Metrics of Classifiers

Model	**Accuracy**	**Precision**	**Recall**	**F1 Score**
LightGBM	0.89	0.84	0.76	0.80
AdaBoost	0.88	0.80	0.75	0.78
Gradient Boosting	0.88	0.80	0.75	0.77

The deployment of the machine learning model on Azure Cloud was successful, with the model now operational and capable of making real-time predictions. The deployment process demonstrated the effectiveness of Azure Cloud services in hosting and managing machine learning models. The ability to monitor and maintain the model ensures its long-term viability and reliability in a production setting.

5.4 Models Feature Importance

LightGBM

In the LightGBM classifier feature importance analysis, tenure in months stands out

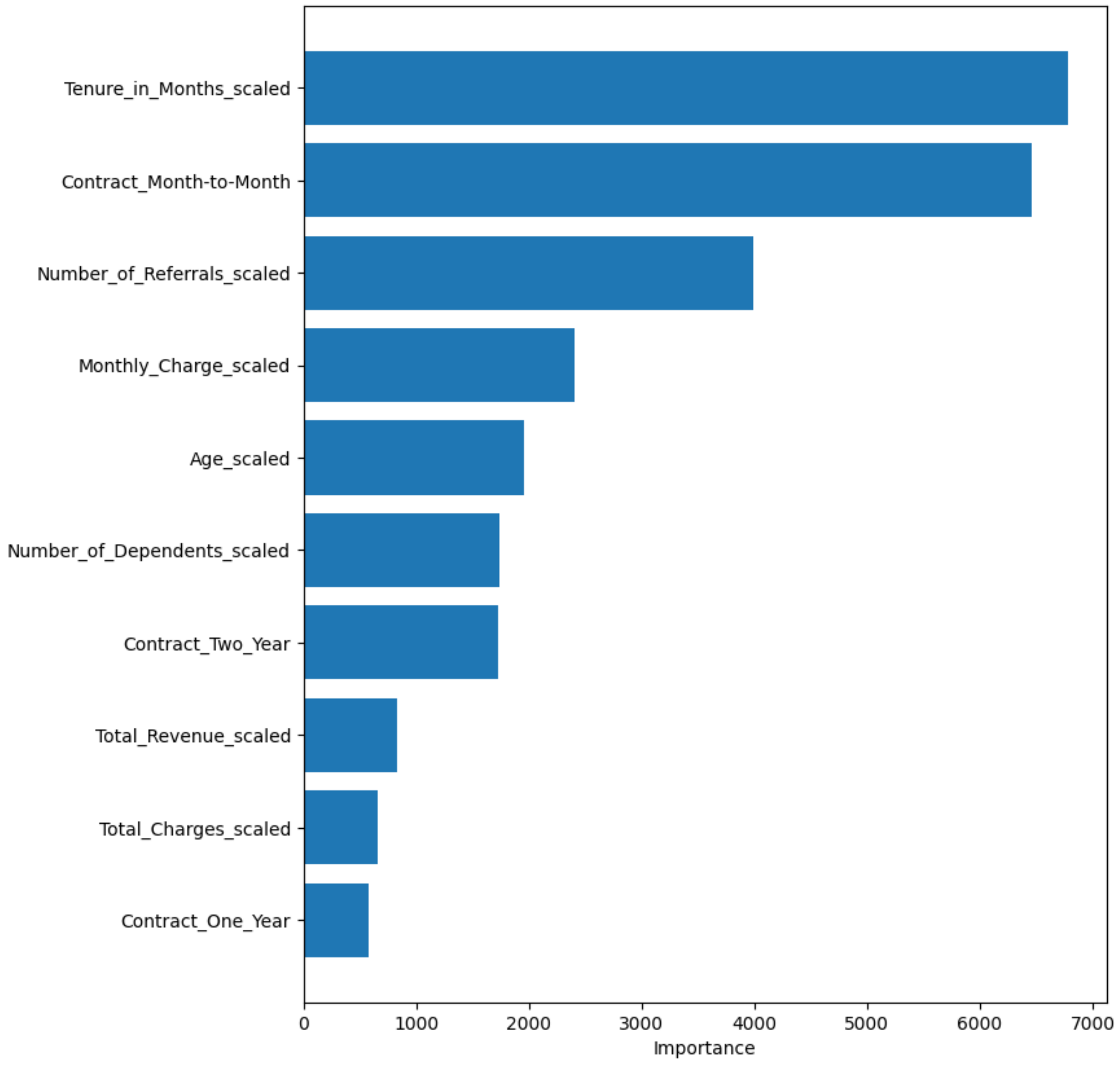

Figure 5.1: Top 10 Feature Importance in LightGBM.

as the most influential feature, which could suggest a strong correlation between the duration of the customer's engagement and their behavior (Figure 5.1). This might indicate that long-standing customers exhibit distinct patterns, potentially due to a higher level of service familiarity or satisfaction. Contract month-to-month is the second-most significant feature, emphasizing the critical nature of flexible, short-term agreements in predicting customer behavior. This most likely reflects the higher churn risk associated with customers who are not subject to long-term commitments and may have a higher propensity for switching services. The third-ranked feature, the number of referrals, points to the importance of customer referrals in influencing the model's predictions, signifying that customers obtained through referrals may have different retention rates or service utilization patterns.

The monthly charge is also a key predictive feature, indicating the impact of the cost of services on customer decision-making. This could be due to a direct association between pricing and the perceived value or affordability of the services offered. Further down the list, age and number of dependents suggest that demographic variables contribute to customer profiling, with potential implications for targeting and customer service strategies. Interestingly, contract two year has a lower degree of importance compared to month-to-month contracts, implying that the long-term nature of these contracts might lead to more stable customer behavior, which is less volatile and thus less indicative in the model.

The last three in the top 10 features of importance in the LightGMB model are total revenue, total charges, and contract one year. Although they do contribute to the model's prediction capability, their contribution is significantly lower when compared to the top five features.

Gradient Boosting

In the Gradient Boosting classifier, the feature importance analysis yields a distinctive hierarchy. Contract month-to-month emerges as the primary feature, signaling the strong influence of short-term contract preferences on the model's output. Tenure in months follows, pointing to the relevance of customer tenure. The number of referrals and monthly charge are also prominent, suggesting the impact of referral programs and billing amounts.

Age appears as a mid-level feature, indicating a moderate role for customer age. The number of dependents implies that family size has a certain level of influence. Total revenue and total charges reflect the financial engagement between the customer and the company.

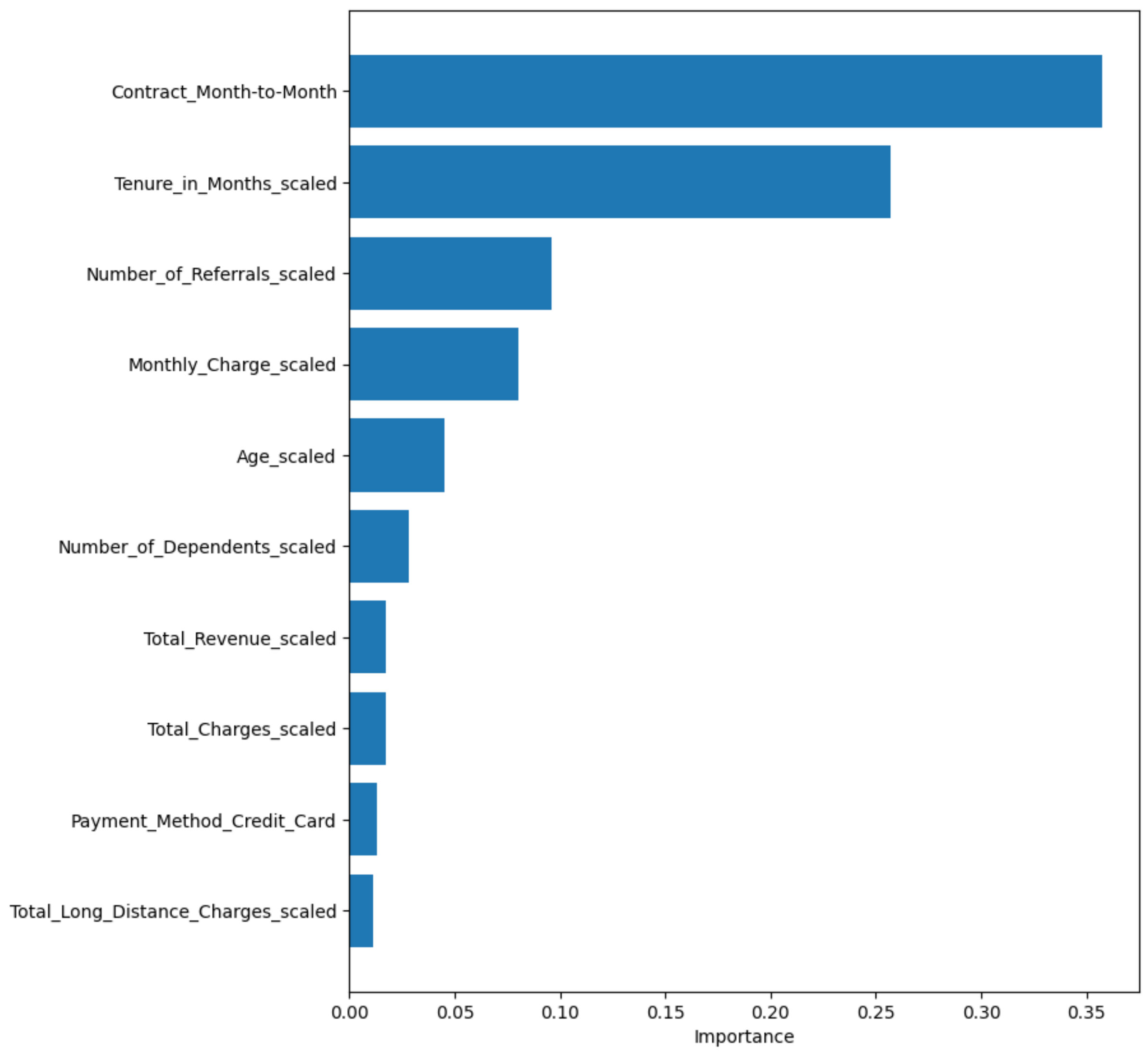

Figure 5.2: Top 10 Feature Importance in Gradient Boosting.

Less prominent but still included are payment method credit card and total long distance charges, denoting the less pronounced but existing influence of payment methods and specific service charges. Together, these features show what the Gradient Boosting model thinks is important for predicting customer behavior. They are different from what the LightGBM model found, but they do overlap.

AdaBoost

The top 10 features influencing the prediction of the AdaBoost model are illustrated

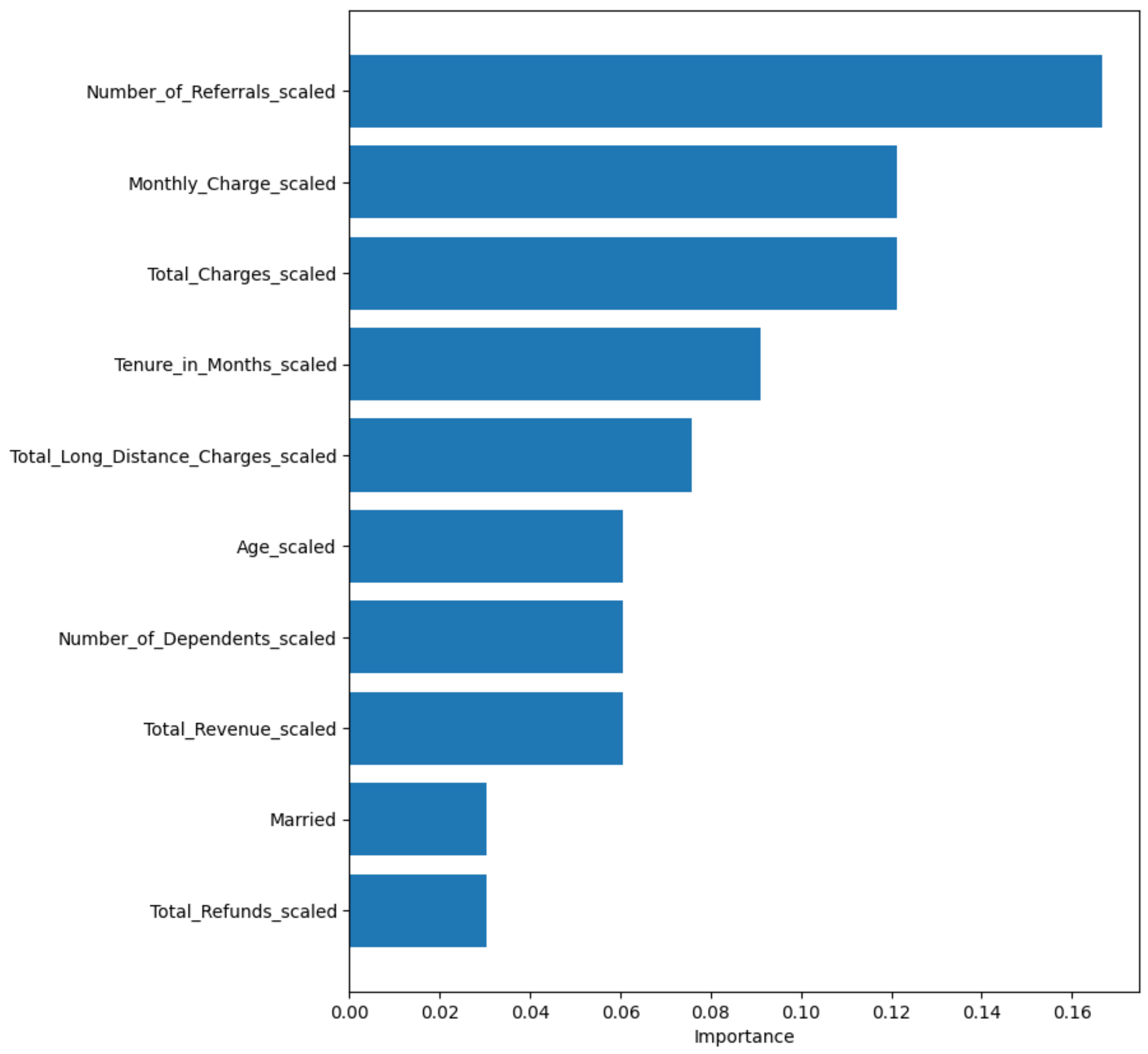

Figure 5.3: Top 10 Feature Importance in AdaBoost.

in Figure 5.3. In the AdaBoost classifier, the number of referrals takes precedence, indicating the significance of customer referrals in predicting outcomes. Monthly charge and total charges follow, signifying the impact of current and cumulative charges on customer behavior. Tenure in months highlights the importance of the duration of service usage.

Total long distance charges is ranked next, pointing to the relevance of specific service costs. Age and the number of dependents suggest that demographic elements, though less dominant, are factored into customer profiling. Total revenue hints at the weight of overall financial contribution in the predictive process.

The lesser yet present influence of marriage suggests a nuanced contribution of social status to customer predictions. Total refunds, while least important in the top ten, complete the feature set, potentially reflecting service satisfaction or resolution effectiveness. This array of features in the AdaBoost classifier presents a comprehensive yet distinct view of the customer profile, emphasizing different aspects of customer engagement compared to the LightGBM and Gradient Boosting models.

Chapter 6

Discussions and Conclusions

6.1 Research Objectives and Approach

The main focus of this research pivots on the pressing issue of customer churn, a phenomenon where customers discontinue their business with a service provider, which in this study pertains to the telecommunications sector. The primary objective has been to delineate and model the myriad factors contributing to churn, with a view to preemptively flag potential churners, thereby enabling targeted customer retention strategies. This goal can be broken down into three parts: finding important churn predictors, looking into the links between customer attributes and churn likelihood, and using machine learning to make a reliable predictive model.

To achieve these objectives, a detailed methodology was adhered to, beginning with an exploratory data analysis of the telecom customer churn dataset acquired from the Maven Analytics website. The dataset, reflecting Q2 2022 data for a customer base of 7043, served as a microcosm of the wider industry context, representing a diverse array of demographic, service usage, billing, and retention-related features. These analyses were very important for figuring out what factors affected behavior and helped with the next step of getting the data ready, which made sure that the base for predictive modeling was strong.

To prevent data leakage and guarantee the integrity of the models' training, the data preparation phase included thorough data cleaning, feature selection, transformation, and stratified dataset splitting. Scaling and encoding numerical and categorical variables, respectively, facilitated the adaptation of the dataset for the machine learning algorithms, which require numerical input. These preparatory steps were critical in addressing the inherent class imbalance through techniques like SMOTE, thus setting the stage for equitable model training.

Many machine learning models were used in predictive modeling, from standard ones like logistic regression, support vector machines, and decision trees to more complex ensemble models like XGBoost and LightGBM. We tested these models using a set of performance metrics, including recall, accuracy, precision, and F1 score. We did this using stratified k-fold cross-validation, which not only gave a good estimate of performance but also avoided the skewed view that comes from having uneven data.

During the model evaluation phase, ensemble classifiers came out on top. This is in line with recent research and industry findings that these algorithms are good at complex pattern recognition tasks like predicting churn. The top three classifiers, LightGBM, Gradient Boosting, and AdaBoost, from fold-fold cross-validation and the unseen test set were selected for further testing. Their performance was fine-tuned through hyperparameter optimization. The focus was to strike a balance between prediction accuracy and model interpretability, a balance that bears significant relevance for actionable business insights.

Overall, the strict methodology used in this study, from early design to model evaluation, was a big part of making sure that the predictive models created were not only correct but also useful and applicable to people in the telecommunications industry. This comprehensive approach made sure that a systematic, data-driven analysis governed the analytics lifecycle from data collection to model deployment, laying the groundwork for effective churn mitigation strategies.

6.2 Model Performance and Feature Importance

The models' evaluation in this study was comprehensive, encompassing a variety of machine learning algorithms, with a specific focus on ensemble methods like LightGBM, Gradient Boosting, and AdaBoost, known for their ability to consolidate weak learners into formidable predictive models. Notably, these ensemble methods had great performance metrics, especially high precision, recall, and F1 scores, which are very important when dealing with churn datasets' imbalanced classes.

The feature importance analysis revealed intriguing insights across the models. For instance, in the LightGBM model, tenure in months' emerged as a critical predictor, which might be indicative of the hypothesis that customer churn is inversely proportional to the duration of the relationship with the provider. This is an intuitive finding, as longer tenure could be associated with higher levels of customer satisfaction or inertia against change.

Contract month-to-month was also highlighted as an important feature. This was particularly picked up on by the Gradient Boosting model. It seems like the type of contract a customer has is important to whether they will churn or not. If a customer has an open-ended contract, this will increase the likelihood of them churning, as there is no cost to them doing so. Companies could pursue any number of initiatives based on this insight, including encouraging their customers to sign up for longer contracts that will benefit them.

In the case of the AdaBoost model, the number of referrals stood out, highlighting the potential role of word-of-mouth and referral programs in customer retention. This suggests that customers who are referred may be more likely to stay due to the social bonds or perceived credibility of the referral source.

The importance of billing-related features like monthly charges and total charges across models indicates a strong correlation between pricing strategy and churn. This could potentially influence a company's pricing strategy, perhaps indicating a need for competitive pricing or tiered services to cater to different customer segments.

Demographic' features like age and number of dependents were less influential but still significant, providing a nuanced understanding of the customer base and implying that personalized marketing strategies could be more effective.

Despite the models' strong predictive performance, they did face challenges. For instance, certain models may struggle with new or unseen data, and the fine-tuning of hyperparameters showed that there is a delicate balance between model complexity and generalizability.

6.3 Interpretation of Results

The models' performance evaluation offers crucial insights into the predictive capabilities of the employed algorithms. The utilization of various classifiers, from basic logistic regression to sophisticated ensemble methods, revealed nuanced understandings of the dataset's complexities. Ensemble methods, with their inherent mechanisms to amalgamate the strengths of numerous weak learners, were observed to excel. This can be attributed to their capacity for capturing non-linear patterns and interactions between features that may not be apparent or may be underrepresented in simpler models.

The classifier's performance was evaluated using stratified k-fold cross-validation, a method picked because it works well with the imbalanced class distribution that comes with churn datasets. The LightGBM, Gradient Boosting, and AdaBoost algorithms emerged as top performers, demonstrating the effectiveness of ensemble methods in this domain. LightGBM's adaptability and speed in managing large datasets and its leaf-wise growth strategy for decision trees allowed for a more refined model that could efficiently handle the feature space. Hyperparameters that regulate the model's complexity and learning rate further strengthened the LightGBM classifier's robustness, which further solidified its suitability for the task at hand.

The feature importance analysis across models provided an interpretive lens for the customer churn phenomenon within the telecommunications sector. Features like tenure in months and contract month-to-month stood out, corroborating industry wisdom that customer loyalty is intricately tied to the length and terms of service. Furthermore, the number of referrals, which ranked prominently in the AdaBoost model, underscores the value of social influence and network effects in customer retention strategies.

6.4 Implications for Practitioners

Strategic Customer Engagement: The study substantiates the assertion that customer engagement strategies can significantly benefit from the application of machine learning insights. The feature importance analysis from models like LightGBM highlights the need for targeted customer engagement practices, especially focusing on tenure and contractual flexibility.

Contractual Adjustments: Findings point to the different implications of month-to-month versus long-term contracts on customer loyalty. Practitioners can use this data to design contracts that better serve customer preferences and foster long-term loyalty.

Referral Programs: The AdaBoost model, in particular, showcased the influence of referral programs on customer retention. This implicates that encouraging existing customers to make referrals can be an effective strategy in enhancing the customer base and reducing churn.

Pricing and Billing Strategies: Ensemble methods have demonstrated the predictive importance of billing-related features. This insight can inform practitioners to revise their pricing strategies to balance affordability and service value, aiming to reduce churn rates associated with billing concerns.

Demographic Personalization: Demographic variables, though less dominant, were still significant in the models, underscoring the importance of personalized marketing and customer service strategies that cater to the specific needs of different customer segments.

Product and Service Development: The importance of specific features like Total Long Distance Charges suggests that service usage patterns are key in predicting churn. This can direct product development teams to create services that more accurately meet customer needs, potentially reducing the incentive to churn.

Retention Campaigns: By understanding the drivers behind churn, marketing teams can tailor retention campaigns more effectively, focusing on the areas most likely to influence customer decisions to stay.

Predictive Customer Support: Deploying the final model into a production environment enables real-time prediction of customer churn. This can be particularly useful for customer support teams, as they could receive alerts about customers who might be at risk of churning, prompting timely intervention.

Policy Interventions: Analysis of churn reasons provides a clear directive for policy change within an organization. Offers from competitors, service dissatisfaction, and pricing are identified as primary churn factors, signaling areas where policy interventions can make a substantial difference.

Technology Integration: The report underscores the importance of integrating machine learning models with existing IT infrastructure. The successful deployment on Azure Cloud demonstrates the practicality and accessibility of such predictive models in a business setting, suggesting that companies should invest in the technology required to integrate advanced analytics into their operational workflow.

6.5 Contribution

The study showcases the effectiveness of different supervised machine learning models, namely ensemble techniques such as LightGBM, Gradient Boosting, and AdaBoost, in accurately forecasting customer attrition. This contributes to the increasing amount of evidence that machine learning offers a strong analytical foundation for dealing with churn.

The research offers valuable insights into the primary aspects that contribute to churn, including the duration of customer retention, the types of contracts, the number of referrals, and the amount of invoicing. This contributes to our comprehension of consumer behavior and assists in the creation of focused retention efforts.

The study demonstrates the potential for real-time churn prediction by implementing the final LightGBM model in a production scenario on the Azure Cloud. This enables firms to promptly identify potential customers who are likely to stop using their services and take proactive actions to keep them.

The research showcases the integration of complex prediction models with smart business methods, emphasizing the harmonious relationship between data science and business operations. This facilitates the implementation of better-informed decision-making and more effective churn management measures.

This study adds to the body of academic literature on churn prediction by showing that ensemble approaches work in a real-world telecoms dataset. This might function as a point of reference for future research on the topic.

6.6 Limitations

The study utilized data from a single telecommunications company in California for a specific quarter. This narrow data scope limits the generalizability of the findings to other regions, times, and industries.

Despite the robust performance of the selected models, they are trained and validated on a specific dataset. Their effectiveness in a real-world scenario across different customer demographics or business models is yet to be tested.

There is a possibility that additional relevant features, not included in the dataset, could improve model performance. Furthermore, the study does not explore feature engineering to a large extent, which could uncover more nuanced relationships.

While hyperparameter tuning was applied to improve model performance, the computational intensity of this process was a limiting factor. Moreover, the search for the optimal parameters may not have been exhaustive due to computing resource constraints.

6.7 Future Perspective

Future research can explore further directions. Here are a few of them:

1. Future studies could incorporate data from multiple service providers, different geographic locations, and span over different time frames to enhance the robustness and applicability of the churn prediction models.

2. Delving deeper into feature engineering and the inclusion of interaction terms might provide new insights and improve predictive accuracy.

3. Applying the churn prediction model to different industries may reveal unique industry-specific churn patterns and validate the model's utility across sectors.

4. Development of systems that learn continuously from new data could maintain the relevance of the model, addressing the issue of concept drift in customer behavior.

5. Creating a user-friendly application or dashboard for non-technical stakeholders could facilitate the adoption of machine learning models in business practices.

Bibliography

Ahmad, A. K., Jafar, A., & Aljoumaa, K. (2019). Customer churn prediction in telecom using machine learning in big data platform. *Journal of Big Data*, *6*(1), 1–24.

Alshdaifat, E., Alshdaifat, D., Alsarhan, A., Hussein, F., & El-Salhi, S. M. F. S. (2021). The effect of preprocessing techniques, applied to numeric features, on classification algorithms' performance. *Data*, *6*(2), 11.

Argesanu, A.-I., & Andreescu, G.-D. (2021). A platform to manage the end-to-end lifecycle of batch-prediction machine learning models. *2021 IEEE 15th International Symposium on Applied Computational Intelligence and Informatics (SACI)*, 000329–000334.

Asbai, N., & Amrouche, A. (2017). Boosting scores fusion approach using front-end diversity and adaboost algorithm, for speaker verification. *Computers & Electrical Engineering*, *62*, 648–662.

Breiman, L. (1996). Bagging predictors. *Machine learning*, *24*, 123–140.

Breiman, L. (2001). Random forests. *Machine learning*, *45*, 5–32.

Bryan, S. (2020). Weighting confusion matrices by outcomes and observations.

Bühlmann, P., & Yu, B. (2002). Analyzing bagging. *The annals of Statistics*, *30*(4), 927–961.

Džeroski, S., & Ženko, B. (2004). Is combining classifiers with stacking better than selecting the best one? *Machine learning*, *54*, 255–273.

Elnagar, A., Al-Debsi, R., & Einea, O. (2020). Arabic text classification using deep learning models. *Information Processing & Management*, *57*(1), 102121.

Elreedy, D., & Atiya, A. F. (2019). A comprehensive analysis of synthetic minority oversampling technique (smote) for handling class imbalance. *Information Sciences*, *505*, 32–64.

Freund, Y., Iyer, R., Schapire, R. E., & Singer, Y. (2003). An efficient boosting algorithm for combining preferences. *Journal of machine learning research*, *4*(Nov), 933–969.

Freund, Y., Schapire, R. E., et al. (1996). Experiments with a new boosting algorithm. *icml*, *96*, 148–156.

Friedman, J., Hastie, T., & Tibshirani, R. (2000). Additive logistic regression: A statistical view of boosting (with discussion and a rejoinder by the authors). *The annals of statistics*, *28*(2), 337–407.

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *Annals of statistics*, 1189–1232.

García, Á. L., De Lucas, J. M., Antonacci, M., Zu Castell, W., David, M., Hardt, M., Iglesias, L. L., Moltó, G., Plociennik, M., Tran, V., et al. (2020). A cloud-based framework for machine learning workloads and applications. *IEEE access*, *8*, 18681–18692.

Geiler, L., Affeldt, S., & Nadif, M. (2022). A survey on machine learning methods for churn prediction. *International Journal of Data Science and Analytics*, *14*(3), 217–242.

Géron, A. (2022). *Hands-on machine learning with scikit-learn, keras, and tensorflow.* " O'Reilly Media, Inc."

Haumahu, J., Permana, S., & Yaddarabullah, Y. (2021). Fake news classification for indonesian news using extreme gradient boosting (xgboost). *IOP Conference Series: Materials Science and Engineering*, *1098*(5), 052081.

Ilić, A. S., Stojanović, D., Stojanović, N., & Ilić, M. (2024). Machine learning model as a service in smart agriculture systems. *Conference on Information Technology and its Applications*, 141–148.

Jain, H., Yadav, G., & Manoov, R. (2020). Churn prediction and retention in banking, telecom and it sectors using machine learning techniques. In *Advances in machine* *learning and computational intelligence: Proceedings of icmlci 2019* (pp. 137–156). Springer.

Joseph, V. R., & Vakayil, A. (2022). Split: An optimal method for data splitting. *Technometrics*, *64*(2), 166–176.

Khodabandehlou, S., & Zivari Rahman, M. (2017). Comparison of supervised machine learning techniques for customer churn prediction based on analysis of customer behavior. *Journal of Systems and Information Technology*, *19*(1/2), 65–93.

Krishnan, K. (2013). *Data warehousing in the age of big data*. Newnes.

Lai, L., & Suda, N. (2018). Rethinking machine learning development and deployment for edge devices. *arXiv preprint arXiv:1806.07846*.

Lalwani, P., Mishra, M. K., Chadha, J. S., & Sethi, P. (2022). Customer churn prediction system: A machine learning approach. *Computing*, 1–24.

Latha, C. B. C., & Jeeva, S. C. (2019). Improving the accuracy of prediction of heart disease risk based on ensemble classification techniques. *Informatics in Medicine Unlocked*, *16*, 100203.

Lin, S., Liu, N., Nazemi, M., Li, H., Ding, C., Wang, Y., & Pedram, M. (2018). Fft-based deep learning deployment in embedded systems. *2018 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 1045–1050.

Mohammed, R., Rawashdeh, J., & Abdullah, M. (2020). Machine learning with oversampling and undersampling techniques: Overview study and experimental results. *2020 11th international conference on information and communication systems (ICICS)*, 243–248.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Qureshi, S. A., Rehman, A. S., Qamar, A. M., Kamal, A., & Rehman, A. (2013). Telecommunication subscribers' churn prediction model using machine learning. *Eighth international conference on digital information management (ICDIM 2013)*, 131–136.

Rahman, M., & Kumar, V. (2020). Machine learning based customer churn prediction in banking. *2020 4th international conference on electronics, communication and aerospace technology (ICECA)*, 1196–1201.

Seger, C. (2018). An investigation of categorical variable encoding techniques in machine learning: Binary versus one-hot and feature hashing.

Shabtai, A., Elovici, Y., Rokach, L., Shabtai, A., Elovici, Y., & Rokach, L. (2012). *Data leakage detection/prevention solutions*. Springer.

Sisodia, D. S., Vishwakarma, S., & Pujahari, A. (2017). Evaluation of machine learning models for employee churn prediction. *2017 international conference on inventive computing and informatics (icici)*, 1016–1020.

Sun, B., Chen, S., Wang, J., & Chen, H. (2016). A robust multi-class adaboost algorithm for mislabeled noisy data. *Knowledge-Based Systems*, *102*, 87–102.

Ullah, I., Raza, B., Malik, A. K., Imran, M., Islam, S. U., & Kim, S. W. (2019). A churn prediction model using random forest: Analysis of machine learning techniques for churn prediction and factor identification in telecom sector. *IEEE access*, *7*, 60134–60149.

Witten, D., & James, G. (2013). *An introduction to statistical learning with applications in r*. springer publication.

Xiong, Y., Ye, M., & Wu, C. (2021). Cancer classification with a cost-sensitive naive bayes stacking ensemble. *Computational and Mathematical Methods in Medicine*, *2021*.

Appendix A

Source Codes

The data and code for this project is stored in the GitHub Repository below,

https://github.com/khanhgeo/DASC6910-GraduateProject/tree/main