

ANTICIPATING CUSTOMER CHURN IN TELECOMMUNICATION USING MACHINE LEARNING ALGORITHM FOR CUSTOMER RETENTION

A PROJECT REPORT

Submitted by

RAJARAJAN K [RA2011003010511]

VISHAL REDDY [RA2011003010501]

Under the Guidance of

Dr. S. Priya

Assistant Professor, Department of Computing Technologies

In partial fulfilment of the requirements for the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING



**DEPARTMENT OF COMPUTING TECHNOLOGIES
COLLEGE OF ENGINEERING AND TECHNOLOGY
SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
KATTANKULATHUR – 603 203**

MAY 2024



**SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
KATTANKULATHUR – 603 203**

BONAFIDE CERTIFICATE

Certified that **18CSP107L** Project Report titled “**ANTICIPATING CUSTOMER CHURN IN TELECOMMUNICATION USING MACHINE LEARNING ALGORITHMS FOR CUSTOMER RETENTION**” is the bonafide work of **RAJARAJAN K [RA2011003010511]** and **VISHAL REDDY [RA2011003010501]** who carried out the project work under my supervision. Certified further, that to the best of my knowledge, the work reported here in does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion for this or any other candidate.

Dr. S. PRIYA
SUPERVISOR
Assistant Professor
Department of Computing Technologies

Dr. T. MANORANJITHAM
PANEL HEAD
Associate Professor
Department of Computing Technologies

Dr. M. PUSHPALATHA
HEAD OF THE DEPARTMENT
Department of Computing Technologies

Internal Examiner

External Examiner



Department of Computing Technologies
SRM Institute of Science and Technology

Own Work Declaration Form

Degree/ Course : B. Tech in Computer Science and Engineering
Student Names : Rajarajan K, Vishal Reddy
Registration Number : RA2011003010511, RA2011003010501
Title of Work : Anticipating Customer Churn in Telecommunication
using Machine Learning Algorithms for Customer
Retention

We hereby certify that this assessment compiles with the University's Rules and Regulations relating to Academic misconduct and plagiarism, as listed in the University Website, Regulations, and the Education Committee guidelines.

We confirm that all the work contained in this assessment is our own except where indicated, and that we have met the following conditions:

Clearly references / listed all sources as appropriate

Referenced and put in inverted commas all quoted text (from books, web, etc.)

Not made any use of the report(s) or essay(s) of any other student(s) either past or present

Acknowledged in appropriate places any help that I have received from others (e.g., fellow students, technicians, statisticians, external sources)

Compiled with any other plagiarism criteria specified in the Course handbook / University website

We understand that any false claim for this work will be penalized in accordance with the University policies and regulations.

DECLARATION:

We are aware of and understand the University's policy on Academic misconduct and plagiarism and we certify that this assessment is our own work, except where indicated by referring, and that we have followed the good academic practices noted above.

Rajarajan K Signature:

Vishal Reddy Signature:

Date: 25-04-24

ACKNOWLEDGEMENT

We express our humble gratitude to **Dr. C. Muthamizhchelvan**, Vice-Chancellor, SRM Institute of Science and Technology, for the facilities extended for the project work and his continued support.

We extend our sincere thanks to Dean-CET, SRM Institute of Science and Technology, **Dr. T. V. Gopal**, for his invaluable support.

We wish to thank **Dr. Revathi Venkataraman**, Professor and Chairperson, School of Computing, SRM Institute of Science and Technology, for her support throughout the project work.

We are incredibly grateful to our Head of the Department, **Dr. M. Pushpalatha**, Professor, Department of Computing Technologies, SRM Institute of Science and Technology, for her suggestions and encouragement at all the stages of the project work.

We want to convey our thanks to our Project Coordinators, Panel Head, **Dr. T. Manoranjitham**, Associate Professor and Panel Members, **Dr. J. Kalaivani**, Associate Professor, and **Dr. S. Vidhya**, Assistant Professor, Department of Computing Technologies, SRM Institute of Science and Technology, for their inputs during the project reviews and support. We register our immeasurable thanks to our Faculty Advisors, **Mrs. R. Brindha**, Assistant Professor, and **Dr. G. Balamurugan**, Assistant Professor, Department of Computing Technologies, SRM Institute of Science and Technology, for leading and helping us to complete our course.

Our inexpressible respect and thanks to our guide, **Dr. S. Priya**, Assistant Professor, Department of Computing Technologies, SRM Institute of Science and Technology, for providing us with an opportunity to pursue our project under her mentorship. She provided us with the freedom and support to explore the research topics of our interest. Her passion for solving problems and making a difference in the world has always been inspiring.

We sincerely thank all the staff and students of Computing Technologies Department, School of Computing, S.R.M Institute of Science and Technology, for their help during our project. Finally, we would like to thank our parents, family members, and friends for their unconditional love, constant support and encouragement.

RAJARAJAN K [Reg. No: RA2011003010511]

VISHAL REDDY [Reg. No: RA2011003010501]

ABSTRACT

The Telecommunication Industry faces significant challenges in maintaining customer loyalty in a highly competitive market. This paper presents an in-depth analysis of customer churn prediction as a strategic tool to enhance customer retention rates. Using machine learning (ML) models, we explore the predictive power of various algorithms on churn and provide actionable insights into customer behavior and retention strategies. We conduct a comprehensive review of existing methodologies, develop and test several predictive models, and evaluate their effectiveness based on real-world data from a major telecommunications provider. Customer attrition is a significant issue and a top priority for large corporations. This study addresses this pressing issue. The dataset is obtained from Kaggle which constitutes of training set and testing set, 80 percent and 20 percent of the entire dataset respectively for identifying customers who tend to unsubscribe in the telecommunications industry. Predicting customer churn with precision is a difficult endeavor, mostly due to the dependence on a singular prediction model in most existing projects. A novel approach is proposed that compares multiple prediction models like logistic regression, KNN, random forest, SVV, Gaussian NB, Kernel SVM, Support Vector Machine, to estimate customer churn. With ROC AUC Mean score of 84% for Logistic Regression produced a better result. In the future, the study might be expanded to investigate the evolving behavioral patterns of churn consumers with the application of Artificial Intelligence approaches for predictive and trend analysis to preserve valuable client.

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	vi
	TABLE OF CONTENTS	vii
	LIST OF FIGURES	viii
	LIST OF TABLES	ix
	LIST OF ABBREVIATIONS	x
1	INTRODUCTION	1
	1.1 Overview	1
	1.1.1 Background and Significance	1
	1.1.2 Research Gap	2
	1.1.3 Objectives of the Research	2
	1.1.4 Methodology	2
	1.1.5 Contributions	3
	1.2 Diverse Categories of Customer Churn	3
	1.3 General Steps Involved in Churn Prediction	4
	1.4 General Objectives	7
2	RELATED WORK	11
3	DESIGN OF ANTICIPATING CHURN PREDICTION MODEL	24
	3.1 System Architecture	24
	3.2 Proposed workflow	29
4	ANALYSIS OF ANTICIPATING CHURN PREDICTION MODEL	31
	4.1 Existing System	31
	4.2 Proposed System	32
	4.3 Implementation of Anticipating customer Churn Prediction Model	39
5	RESULT AND DISCUSSION	47
6	CONCLUSION AND FUTURE ENHANCEMENT	51
	6.1 Conclusion	51
	6.2 Future Enhancement	52
	REFERENCES	53
	APPENDIX 1	55
	PLAGIARISM REPORT	92
	PUBLICATION	93

LIST OF FIGURES

FIGURE NO.	TITLE	PAGE NO.
3.1	Architecture of the anticipating churn prediction model	24
4.1	Histogram Of Numerical Data	39
4.2	Distribution of Label Encoded Categorical Variables	41
4.3	Correlation with Churn Rate	42
5.1	Confusion Matrix	48
5.2	ROC Graph	48

LIST OF TABLES

TABLE NO.	TITLE	PAGE NO.
4.1	Comparative Analysis of various classification techniques	43
5.1	Comparing the accuracies of different algorithms	47
5.2	Comparison of Baseline Classification Algorithms –2 nd Iteration	47
5.3	Determining churn rate with prediction of propensity score – high risk	50
5.4	Determining churn rate with prediction of propensity score – low risk	50

LIST OF ABBREVIATIONS

ANN	Artificial Neural Network
CNN	Convolutional neural network
MLP	Multilayer perception
MFCC	Mel-Frequency Cepstral Coefficients
KNN	K-nearest neighbor
SVM	Support Vector Machine

CHAPTER 1

INTRODUCTION

1.1 Overview

In today's highly competitive telecommunication sector, the ability to retain customers is as crucial as attracting new ones. Customer churn, which refers to the loss of clients or customers, directly impacts a company's revenue and long-term sustainability. The advent of machine learning (ML) has introduced innovative approaches to predict customer churn, allowing telecommunication companies to implement proactive strategies for retention. This research paper aims to explore the effectiveness of machine learning models in predicting customer churn, providing insights that could lead to enhanced customer retention strategies in the telecommunication sector.

1.1.1 Background and Significance

The telecommunication industry is characterized by its rapid pace of technological advancements and the high level of competition. Customers have a variety of choices, making them more prone to switching service providers for better quality, lower prices, or enhanced services. According to industry reports, the average churn rate for telecommunications companies is significantly higher than in many other industries, highlighting the need for effective churn prediction models. The cost implications of losing customers are substantial, not just in terms of lost revenue but also the increased marketing costs required to replace them. Therefore, understanding the predictors of churn and effectively forecasting risk can transform a company's strategic approaches to customer retention.

Churn prediction in the telecommunications sector is particularly challenging due to the vast amount of customer data that includes call behavior, service usage, billing information, customer service interactions, and social media feedback. Traditional analytical techniques often fail to handle the complexity and scale of this data, which is where machine learning models come into play. These models can learn from large datasets and identify subtle patterns that may indicate an impending churn, making them invaluable tools for customer retention.

1.1.2 Research Gap

While numerous studies have explored the use of machine learning in churn prediction, there is a continual need for research that connects these predictions more closely with actionable retention strategies. Many existing models focus primarily on predicting churn without providing clear insights into the specific factors influencing customer decisions at a granular level. Additionally, the dynamic nature of customer preferences and the introduction of new services and technologies demand ongoing refinement and testing of these models. There is also a significant gap in the literature regarding the comparison of various machine learning techniques and their practical implementation in real-world scenarios.

1.1.3 Objectives of the Research

This study aims to fill these gaps by achieving the following objectives:

- To critically review existing machine learning models used for churn prediction in the telecommunication sector.
- To develop and test several machine learning models, including logistic regression, decision trees, support vector machines, and neural networks, to compare their effectiveness in predicting customer churn.
- To identify the most significant predictors of churn in the telecommunications industry and understand how these vary across different customer segments.
- To integrate the insights from machine learning models into practical strategies for customer retention.
- To evaluate the impact of these strategies on improving customer retention rates and reducing churn.

1.1.4 Methodology

The methodology of this study involves several key phases:

- i. **Data Collection:** Gathering a comprehensive dataset from a leading telecommunications company, which includes demographic data, service usage patterns, billing history, customer service interactions, and social media sentiment.
- ii. **Data Preprocessing:** Cleaning the data to handle missing values, encoding categorical variables, and normalizing data to prepare it for analysis.
- iii. **Feature Selection:** Using statistical methods and machine learning algorithms to

identify and select features that have the most significant impact on churn.

- iv. **Model Development and Training:** Building various machine learning models and training them on processed data.
- v. **Model Evaluation:** Comparing the models based on accuracy, precision, recall, F1-score, and ROC-AUC values.
- vi. **Implementation of Findings:** Proposing strategic actions based on model outcomes and collaborating with stakeholders to implement retention strategies.

1.1.5 Contributions

This research aims to contribute to both academic knowledge and practical applications. Academically, it aims to enhance the understanding of how different machine learning models can be effectively used in churn prediction. Practically, it seeks to provide telecommunication companies with actionable insights into customer behavior, enabling them to tailor their customer retention strategies more effectively. By integrating machine learning into their operational strategies, companies can not only reduce churn rates but also enhance overall customer satisfaction and loyalty. In conclusion, as the telecommunication sector continues to evolve, the ability to predict and manage customer churn through machine learning will remain a key competitive advantage. This research aims to advance the field by providing a comprehensive analysis of machine learning techniques for churn prediction and their practical application in customer retention strategies.

1.2 Diverse Categories of Customer Churn

Predicting customer churn with precision is a difficult endeavor, mostly due to the dependence on a singular prediction model in most existing projects. To address this, this study proposes a novel approach that compares multiple prediction models like logistic regression, KNN, random forest, SVV, Gaussian NB, Kernel SVM, Support Vector Machine, to estimate customer churn. To reduce churn, businesses must prioritize endeavors aimed at enhancing the customer experience, delivering exceptional customer service, offering competitive pricing, and consistently engaging with customers. The discernment of churn patterns also empowers businesses to pinpoint areas of vulnerability and enact strategic enhancements to retain customers.

Diverse categories of customer churn exist, including:

- **Voluntary churn:** This occurs when customers decide to cut ties with a company, usually due to dissatisfaction with the product or service, higher prices, or more enticing offers from other companies.
- **Involuntary churn:** This happens when a customer is lost due to circumstances beyond their control, such as relocation, death, or changes in personal circumstances.
- **Deliberate churn:** This occurs when a customer intentionally breaks ties with a company, usually with the intention of switching to a competitor or a different product or service.
- **Indirect churn:** This takes place when customers are lost due to the actions of third parties such as resellers, distributors, or affiliates.
- **Passive churn:** This happens once customers stop using a product or service but do not formally cancel their subscription or disengage from the company. Forgetfulness, waning interest, or time constraints can all be reasons for this.

Understanding the various types of churn enables business to develop strategies for reducing customer churn and retaining customers in the long run.

1.3 General Steps Involved in Churn Prediction

1. Data Collection Process

The first stage is the careful collection of customer information covering a range of topics, including demographics, previous transactions, and customer interactions. We have used a dataset from Kaggle that represents a telecom service provider for our example. This methodical approach to gathering data centers on formulating a research question or hypothesis, choosing a representative sample, and choosing the right instruments and techniques for gathering data.

2. Data Preprocessing Steps

After the data is gathered, it undergoes extensive preprocessing. This means dealing with outliers, fixing inconsistencies, and addressing missing values. To make the data suitable for use in machine learning algorithms, it must be transformed and normalized. Data preprocessing is an extensive process with multiple components:

- **Data Cleaning:** removing or correcting data that is erroneous, unnecessary, or incomplete.
- **Data Transformation:** Changing the format of data involves scaling numerical

attributes within a range and converting variables with categories to numerical values.

- Data Reduction: lowering the amount of data by concentrating on important characteristics or sampling a subset.
- Data Integration: combining data from various datasets to create a coherent whole.
- Data Normalization: Scaling data values to achieve a consistent range.
- Data Discretization: dividing up continuous data into distinct groups.

3. Feature Engineering Endeavors

The goal of this phase is to identify relevant features from the data that could have an impact on customer attrition. Variables like customer tenure, frequency of purchases, complaints, and customer satisfaction ratings may be included in these features. Feature engineering is the process of turning raw data into meaningful features for machine learning models. This process encompasses multiple crucial elements:

- Feature Selection: Selecting the most important features from a larger collection, frequently by evaluating the significance of an attribute or doing statistical analysis.
- Feature Extraction: Creating new features out of the ones that already exist by using methods like principal component analysis or domain-specific expertise.
- Feature Scaling: Making sure that feature values, which are necessary for some machine learning models, are standardized to a similar range.
- Feature Normalization: In order to distribute feature values according to a Gaussian distribution, frequently achieved by means of techniques such as the Box-Cox Transformation.

4. Model Selection Considerations

Selecting an appropriate machine learning model for the particular issue at hand is a crucial decision. Several models are used for churn prediction, including logistic regression, decision trees, random forests, and support vector machines. A critical stage in the process is selecting the best model from a pool of candidates that have all been trained on the same dataset. Finding the model that can produce accurate predictions and generalize well to new data is the goal. Techniques for selecting models include holdout validation, bootstrapping, cross-validation, and more. The importance of careful selection is highlighted by the significant influence of model choice on predictive performance.

5. Model Training

The selected model is trained using the preprocessed data. Model training, or teaching the model to make accurate predictions, is an essential part of machine learning. In order for the model to recognize patterns in the incoming data and produce the desired outcomes, it must be trained iteratively with different parameters. The training procedure makes use of a training dataset that contains input feature pairs and matching target labels. The model optimizes its parameters to enable accurate predictions on unseen data based on this data. Training performance is influenced by the hyper-parameters, the algorithm, the optimization techniques, and the quality of the data. By assessing the model's performance with validation techniques like cross-validation, accurate predictions can be guaranteed.

6. Model Evaluation Procedures

A few examples of suitable assessment metrics that are used to assess the performance of the trained model are recall, accuracy, precision, and F1 score. Model assessment, or gauging a trained model's performance on new, untested data, is an essential component of machine learning. The aim is to confirm that the model can produce accurate predictions and generalize well. Evaluation criteria like accuracy, precision, recall, and F1 score are used, depending on the kind of application. Preprocessing data and modifying hyper-parameters are often required for iterative model assessment in order to enhance performance. It also helps with model comparison and choosing the best model for a particular application.

7. Model Deployment and Integration

A model is prepared for deployment in a production environment after it has been trained and assessed. The learned model must be integrated into an end-user-accessible system or application as part of this deployment process. In the deployment environment, performance, reliability, and security optimization might be required.

This entails merging the model with the application's codebase and transforming it into a format that works with the deployment environment. Sustaining accuracy and efficacy requires constant observation and updating.

1.4 General Objectives

Creating a detailed list of objectives for a research work titled "Customer Churn Prediction Using Machine Learning Algorithms" is essential for guiding the research and providing a structured approach to your investigation.

- *To Understand and Define Customer Churn:*
 - Define customer churn specifically in the context relevant to the industry being studied (e.g., telecommunications, finance, etc.).
 - Discuss the importance of predicting customer churn and its impact on business performance.
 - Thoroughly define customer churn within the specific industry context, highlighting different churn types (voluntary vs involuntary).
 - Explore the various factors that contribute to churn, highlighting how they differ across industries and customer segments.
- *To Review Existing Literature:*
 - Conduct a comprehensive literature review on existing models and methods used for customer churn prediction.
 - Identify gaps in current research that could be addressed with newer or different machine learning techniques.
 - Analyze studies that have applied traditional statistical methods alongside newer machine learning approaches to understand the evolution in the field.
 - Identify common challenges and solutions discussed in existing works, such as overfitting, imbalanced datasets, and the interpretability of machine learning models.
- *To Select Relevant Machine Learning Algorithms:*
 - Identify and describe various machine learning algorithms commonly used for churn prediction (e.g., logistic regression, decision trees, support vector machines, neural networks).
 - Assess the advantages and limitations of each algorithm in the context of churn prediction.
 - Discuss in detail the theoretical underpinnings of each algorithm and why certain algorithms might be better suited to types of data or industry conditions.

- Consider the scalability of algorithms and their ability to handle large, complex datasets.
- *To Develop a Methodological Framework:*
 - Establish a methodological framework for applying machine learning algorithms to predict customer churn.
 - Include data collection, data preprocessing, feature selection, model building, and validation steps.
 - Elaborate on the rationale for each step in the machine learning workflow, from data collection to model validation.
 - Discuss the tools and software that will be used for implementing the machine learning models, including any specific libraries or frameworks.
- *To Perform Data Collection and Preprocessing:*
 - Outline the process of collecting relevant customer data from available sources.
 - Describe the preprocessing steps necessary for preparing data for analysis, such as handling missing values, normalization, and encoding categorical variables.
 - Detail the sources of data, whether internal (CRM systems, transaction logs) or external (social media, third-party demographic data).
 - Explain the techniques for dealing with common data issues, such as imbalanced classes in churn prediction, which can bias the model performance.
- *To Identify and Select Features:*
 - Determine which features (customer demographics, transaction history, product usage, etc.) are most predictive of churn.
 - Use techniques like feature importance and selection algorithms to refine the feature set.
 - Use statistical tests and data visualization to justify the selection of predictive features.
 - Explore advanced feature engineering techniques, such as interaction effects between variables and polynomial features, to enhance model performance.
- *To Train and Test Models:*
 - Detail the process of training machine learning models using the selected features.

- Discuss the split of data into training, validation, and test sets.
- Describe the use of different data partitions (e.g., 70-15-15 for train-validation-test splits) to optimize model parameters and prevent overfitting.
- Explain the rationale behind using different training techniques, such as batch training or online learning, in the context of available data.
- *To Evaluate Model Performance:*
 - Implement various metrics to evaluate the performance of each model, such as accuracy, precision, recall, F1-score, and AUC-ROC.
 - Compare the performance of different algorithms to determine the most effective approach.
 - Discuss the trade-offs between different evaluation metrics, especially in scenarios where the cost of false negatives (failing to identify a customer at risk of churning) differs significantly from false positives.
 - Consider the use of advanced statistical techniques to measure model confidence and uncertainty.
- *To Implement Cross-Validation Techniques:*
 - Apply cross-validation techniques to assess the robustness of the predictive models.
 - Explore the use of ensemble techniques to improve prediction accuracy.
 - Provide a detailed explanation of how cross-validation can be leveraged to ensure that the model's performance is consistent across different subsets of the data.
 - Discuss the use of bootstrapping and other resampling techniques to estimate model stability and reliability.
- *To Discuss Ethical and Privacy Considerations:*
 - Address potential ethical and privacy concerns related to the use of customer data in churn prediction.
 - Propose measures to protect customer data and ensure compliance with relevant data protection regulations.
 - Delve into the legal frameworks surrounding data usage, such as GDPR in Europe and CCPA in California, and their implications for customer data analysis.
 - Discuss ethical considerations in predictive modeling, such as the potential for bias in algorithmic decisions and how to mitigate these risks.

- *To Provide Practical Applications and Recommendations:*
 - Translate the research findings into actionable business strategies to reduce customer churn.
 - Recommend specific interventions or changes in business practices based on model insights.
 - Link model outputs to strategic business decisions, such as personalized marketing campaigns or targeted customer retention programs.
 - Evaluate the feasibility and cost-effectiveness of implementing different churn prevention strategies based on model predictions.

- *To Explore the Use of Advanced and Emerging Techniques:*
 - Investigate the potential of advanced machine learning techniques like deep learning or reinforcement learning for churn prediction.
 - Explore the feasibility of integrating alternative data sources (social media sentiment, customer service interactions) to enhance prediction accuracy.
 - Examine cutting-edge developments in machine learning, such as the use of autoencoders for unsupervised feature learning or the application of graph neural networks to leverage relationship data.
 - Propose a framework for integrating unconventional data sources into the churn prediction models, assessing their additional value and challenges.

These objectives, when well-executed, not only structure this research work but also ensure that it contributes valuable insights into the field of customer churn prediction using machine learning. Each of these expanded objectives contributes to a robust, detailed, and scientifically rigorous approach to predicting customer churn using machine learning, ensuring that the research is both comprehensive and applicable in real-world settings.

CHAPTER 2

RELATED WORK

Nyoman Mahayasa Adiputra et al. focused on enhancing the accuracy of predicting customer churn through an ensemble machine learning approach [1]. The authors propose a model that combines various machine learning techniques using a weighted average method, aiming to leverage the strengths of each individual predictor to improve overall prediction performance. The ensemble method discussed in the paper integrates different predictive models and adjusts their contributions through a weighting system. This system assigns more influence to the models that perform better, thereby theoretically increasing the robustness and reliability of the predictions. The study is particularly significant as customer churn prediction is crucial for businesses looking to retain customers and reduce turnover costs. While specific details on the model's architecture, the datasets used, and the exact results of the model's performance were not available in the summary, it's clear that the research adds to the growing body of knowledge on applying advanced ensemble techniques in practical business applications like churn prediction. This method is part of a broader trend in machine learning and data science that seeks to improve decision-making processes in customer relationship management.

In the year 2023, Diao Azzam et al. presented 5th Novel Intelligent and Leading Emerging Sciences Conference (NILES) [2]. This research focuses on integrating the Apriori algorithm with ensemble learning techniques to predict customer churn more effectively. The Apriori algorithm, typically used for mining frequent item sets and association rule learning in databases, is employed here to identify strong rules and patterns in customer behavior that signify churn. The authors' approach leverages ensemble learning, which combines multiple models to improve prediction accuracy compared to individual models. This method usually results in more robust and accurate predictions by aggregating the outcomes of various models and thus reducing the risk of an erroneous prediction by any single model. Although the paper's abstract does not provide detailed results or methodologies, it's evident that the study contributes to the field by exploring the synergy between a traditional data mining algorithm and modern ensemble techniques to tackle the practical problem of customer churn. This kind of research is crucial for businesses looking to enhance their customer retention strategies through advanced predictive analytics.

Yunjie Lin et al. published in 2022, explores advanced techniques in machine learning to enhance the prediction accuracy of customer churn, particularly in the financial sector. This research integrates a bidirectional long short-term memory (BiLSTM) network with convolutional neural networks (CNNs) to form a combined model that addresses the weaknesses of each approach when used separately. This hybrid model, called BiLSTM-CNN, aims to effectively capture both temporal dependencies and spatial hierarchies in data [3]. The study introduces an attention mechanism into the BiLSTM-CNN model, creating the AttnBiLSTM-CNN, which seeks to further refine the model's focus on relevant features for churn prediction. The results indicate slight but noteworthy improvements in various performance metrics such as accuracy, churn rate prediction, F1 score, and AUC values over the base BiLSTM model. These enhancements suggest that the attention mechanism successfully amplifies crucial signals in the data that are predictive of churn. This research underscores the potential of combining multiple neural network architectures to improve predictive performance in customer churn applications, offering a promising tool for financial institutions to proactively manage customer retention and enhance competitive advantage.

The authors Parth Pulkundwar et al. have evaluated various machine learning algorithms to identify the most effective method for predicting customer churn. The study is vital as customer churn prediction helps businesses to develop strategies for customer retention, which is crucial for maintaining revenue and profitability [4]. In their research, the authors compared multiple supervised machine learning models, including decision trees, random forests, support vector machines, and neural networks. Each model was assessed based on its accuracy, precision, recall, and F1 score to determine which algorithm performs best in predicting whether a customer will leave a service or product. This kind of comparative analysis is instrumental in guiding businesses to choose the appropriate predictive analytics tools for their specific needs. The findings of this study contribute to the ongoing dialogue in the field of predictive analytics by highlighting the strengths and weaknesses of each algorithm in the context of churn prediction. By doing so, it provides a clear guideline for companies on optimizing their analytics strategies to reduce churn effectively, thereby enhancing customer satisfaction and loyalty. This research adds to the growing body of knowledge on applying machine learning techniques in practical business applications, particularly in areas requiring predictive insights to foster decision-making and strategic planning.

Eunjo Lee et al. focuses on churn prediction for profitable, long-term customers in online gaming. The authors develop a method that not only predicts churn but also emphasizes maximizing profit by retaining the most valuable players. This approach differentiates from typical churn predictions by concentrating on high-value customers who impact revenue significantly, rather than simply predicting churn across all user levels [5].

Xin Hu et al. explores the development of a hybrid model combining decision trees and neural networks to predict customer churn more effectively [6]. The study emphasizes the benefits of integrating these two approaches, leveraging the decision tree's ability to handle categorical data and interpretability alongside the neural network's prowess in recognizing complex patterns in large datasets. The authors specifically focus on the synergy between the structured, rule-based processing of decision trees and the deep learning capabilities of neural networks to enhance predictive accuracy. This combination allows for a robust model that can handle various data types and structures, making it particularly suitable for the complex and often heterogeneous data environments found in big data scenarios. Empirical tests conducted in the study demonstrate that the combined model outperforms individual models built solely on decision trees or neural networks. This improved performance underscores the potential of hybrid approaches in predictive analytics, particularly in sectors like telecommunications where customer retention is critical. The paper not only provides a detailed comparison of model performances but also discusses the implications of such hybrid models in operational settings, suggesting that they can significantly enhance decision-making processes related to customer management strategies.

In their 2023 study presented, Harish A. S. et al. explored advanced methods for predicting customer churn, specifically comparing cluster-based models against traditional RFM (Recency, Frequency, Monetary) models [7]. The paper evaluates the effectiveness of leveraging clustering techniques, such as K-means, to segment customers into distinct groups based on their behaviors and transaction patterns before applying predictive models to assess churn likelihood. This approach is analyzed in contrast to the conventional use of RFM scoring, which categorizes customers solely on their most recent purchase, frequency of purchases, and total monetary contribution.

The authors detail their methodology by first segmenting the customer data into clusters, then applying machine learning techniques to predict churn within each cluster. The

predictive accuracy of this cluster-based approach is then benchmarked against the RFM-based model. Their findings suggest that the cluster-based approach significantly enhances the predictive power by capturing more complex patterns in customer behavior that RFM scores might overlook. This improvement is quantified through various performance metrics such as accuracy, precision, recall, and F1-score, demonstrating the superiority of the cluster-based method in identifying potential churners more effectively. Harish A. S. and Malathy C. conclude that integrating clustering algorithms into churn prediction frameworks can provide businesses with a more nuanced understanding of customer behavior, leading to more effective customer retention strategies. They advocate for the adoption of these advanced analytical techniques in customer relationship management systems, emphasizing that such approaches can enable more personalized and proactive interventions to reduce churn. The paper's findings have significant implications for businesses looking to enhance their analytical capabilities to retain customers and maximize lifetime value.

Kiran Deep Singh et al. explores customer churn in the telecommunications industry through the lens of exploratory data analysis (EDA) and predictive modeling. The authors provide a comprehensive examination of customer data to unearth underlying patterns and influential factors leading to customer attrition [8]. The primary objective of their study is to utilize these insights to enhance the accuracy and efficacy of churn prediction models, which are crucial for strategic customer retention efforts. The research starts with a detailed EDA, which involves scrutinizing various customer demographics, service usage patterns, and other relevant variables that might affect churn. This initial phase helps in identifying significant predictors of churn and in cleansing the dataset for optimal model performance. The authors employ various visualization techniques to illustrate the distribution and relationship of these factors, thus providing a clear foundation for the subsequent predictive modeling phase. For the predictive analysis, the authors apply several machine learning algorithms, including logistic regression, decision trees, and random forests, to develop a robust churn prediction model. The paper evaluates these models based on their accuracy, precision, recall, and F1-score, demonstrating that ensemble methods like random forests perform best in handling the complex and varied nature of telecom data. The research concludes by discussing the implications of their findings for telecom operators, emphasizing the importance of targeted customer retention strategies and personalized marketing efforts to reduce churn. The paper advocates for ongoing refinement of predictive

models as customer behavior and market dynamics evolve.

In 2022, Mustafa Büyükkeçeci et al. delve into the application of data mining techniques for predicting customer churn. Their research targets a pressing issue in customer relationship management, particularly within industries experiencing high customer turnover [9]. The focus of their study is to develop an effective predictive model using a range of data mining methods to identify customers likely to churn, allowing businesses to implement preventative measures proactively. The authors describe the process of setting up their predictive model, starting with the selection and preparation of relevant customer data, which includes demographic information, service usage patterns, and customer service interactions. They apply various preprocessing techniques to clean and standardize the data, ensuring that the input variables are suitable for effective modeling. The study explores several data mining techniques, including decision trees, support vector machines (SVM), and neural networks, comparing their performance in accurately predicting churn.

The paper concludes with a comprehensive analysis of the results obtained from each model, highlighting the strengths and weaknesses of each approach. The findings indicate that neural networks provided the most accurate predictions among the tested models, though at a cost of increased computational complexity. Büyükkeçeci and Okur emphasize the importance of choosing the right model based on the specific needs and constraints of the business, including factors like available computational resources and the need for model interpretability. Their study contributes valuable insights into the application of advanced data mining techniques to improve customer retention strategies, demonstrating the practical benefits of predictive analytics in a competitive business environment.

In the Conference paper, S. Venkatesh et al. introduce an innovative approach to predicting customer churn in cloud-based services by integrating a genetic algorithm with a support vector machine (SVM) [10]. This study aims to leverage the strengths of both techniques: the genetic algorithm for optimizing the feature selection process and SVM for its robustness in classification tasks. The primary goal is to improve prediction accuracy and efficiency, which is crucial for cloud service providers looking to enhance customer retention and minimize churn. The methodology begins with the application of the genetic algorithm to select an optimal subset of features from a large dataset, which includes user activity, subscription details, service usage patterns, and customer feedback. This optimization is crucial because it reduces the dimensionality of the data, which can otherwise overwhelm

the SVM and lead to overfitting or underperforming models. After the feature selection, these optimized data subsets are used to train the SVM, which classifies customers into churners and non-churners. The authors detail the parameters and iterations used in the genetic algorithm to ensure the best possible feature combination for SVM performance.

The paper concludes with a detailed evaluation of the model's effectiveness, comparing it against traditional churn prediction models that do not use genetic algorithms for feature selection. The results demonstrate that the hybrid approach significantly outperforms standard methods in terms of accuracy, precision, and recall. Venkatesh and Jeyakarthic's research highlight the potential of combining genetic algorithms with machine learning techniques like SVM to enhance predictive analytics capabilities in the cloud services sector. The study not only provides a roadmap for effectively implementing these techniques but also opens the door for future innovations in predictive model optimization.

In the 2023 paper, Aishwarya H M et al. presented at the 4th International Conference on Communication, Computing and Industry 6.0, who explores an innovative approach to customer churn prediction using the Synthetic Minority Oversampling Technique (SMOTE). This study addresses the common issue of class imbalance in churn datasets, where the number of customers who do not churn vastly outnumbers those who do. SMOTE is utilized to artificially augment the minority class (churned customers) by generating synthetic examples, thereby providing a balanced dataset that allows for more effective training of predictive models [11]. The authors detail their methodology beginning with the application of SMOTE to preprocess the data, followed by the use of various machine learning models to predict churn. These models include logistic regression, decision trees, and random forests, each evaluated for its effectiveness in handling the newly balanced dataset. The paper emphasizes the improvement in model performance due to the better representation of the minority class, which often carries more significant predictive power for churn. The results of the study are thoroughly analyzed, showing that the models trained on the SMOTE-enhanced dataset not only improved in accuracy but also in other metrics like precision, recall, and F1-score, which are crucial for evaluating the success of predictive models in practical applications. The research concludes with discussions on the implications of using SMOTE in churn prediction, particularly its potential to enhance the predictability of customer behaviors and facilitate more proactive retention strategies. The paper highlights the importance of addressing class imbalance in predictive modeling within

industries where customer retention is critical.

Abhishek Gaur et al. investigate customer churn in the telecommunications sector using various machine learning techniques [12]. The study is driven by the industry's high churn rates and the significant financial implications associated with losing customers. Gaur and Dubey aim to leverage machine learning to develop predictive models that can accurately identify customers at high risk of churning, thereby enabling telecom companies to implement effective retention strategies. The methodology of the study involves the collection and preprocessing of telecom data, which includes customer demographics, service usage, billing information, and customer service interactions. The authors explore a range of machine learning algorithms, such as logistic regression, decision trees, random forests, and support vector machines, to find the most effective approach for predicting churn. Each model is trained and tested on the dataset, and performance metrics such as accuracy, precision, and recall are used to evaluate and compare their effectiveness. The paper concludes that ensemble methods, particularly random forests, outperform other models in predicting customer churn in the telecom sector. Gaur and Dubey highlight the advantages of using these techniques, including their ability to handle large and complex datasets while providing insights into the importance of various predictive factors. The study underscores the potential of machine learning to transform customer retention strategies in telecommunications by enabling more targeted and timely interventions based on predictive analytics.

Mohamed Galal et al. focus on enhancing customer churn prediction in the digital banking sector through the use of ensemble modeling. This approach is particularly significant in the context of digital banking, where customer retention is critical due to the competitive nature of the industry and the high cost of acquiring new customers. The authors propose a sophisticated ensemble modeling technique that combines multiple machine learning models to improve the accuracy and robustness of churn predictions [13].

The research methodology involves an extensive analysis of customer data collected from digital banking platforms. This data includes variables such as account usage, transaction history, customer demographics, and interaction with digital banking services. The authors utilize a variety of machine learning algorithms—namely decision trees, logistic regression, and gradient boosting machines—to create individual predictive models. These models are

then combined into an ensemble framework using techniques such as bagging and boosting, which are designed to reduce variance and bias, thereby enhancing predictive performance. The paper concludes with a comparative analysis of the predictive accuracy of the ensemble model against the individual models. The results clearly demonstrate that the ensemble approach significantly outperforms the single model predictions in terms of accuracy, precision, and recall. Galal, Rady, and Aref emphasize the importance of using ensemble models in digital banking to effectively predict customer churn, which can enable banks to implement more effective retention strategies. The study highlights the potential of advanced machine learning techniques in helping banks mitigate the risks associated with customer churn.

Fulu Chen et al. explore the use of a stacking model to predict customer churn. This study addresses the challenge of customer retention by leveraging a stacking ensemble approach, which integrates multiple prediction models to enhance the predictive accuracy. The primary innovation of this research lies in its combination of various base learners to form a comprehensive view that captures different aspects of customer behavior and tendencies to churn. The methodology of the study involves initially training several base models, including decision trees, support vector machines, and neural networks, on a dataset comprising customer demographic data, transactional behavior, product usage, and customer service interactions. These models are chosen for their diverse strengths and biases in handling different types of data and patterns. The outputs of these models are then used as inputs into a final meta-model, typically a logistic regression in this study, which aims to synthesize the individual predictions into a final decision. This stacking technique is designed to leverage the unique strengths of each model and offset their individual weaknesses. The results presented in the paper show that the stacking model outperforms each of the individual models when evaluated on standard metrics like accuracy, precision, recall, and AUC (Area Under the Curve). Chen et al. provide a detailed analysis of the performance improvements, attributing them to the meta-model's ability to effectively combine and optimize the diverse predictions. The research concludes with a discussion on the practical implications of using stacking models for churn prediction, particularly in dynamic and competitive industries like telecommunications and banking. The authors suggest that such models can significantly improve strategic decision-making in customer retention initiatives, offering a more reliable tool for identifying at-risk customers and tailoring intervention strategies [14].

Rezaeian et al. explore customer churn prediction using data mining techniques specifically applied to an Iranian payment application [15]. The study focuses on identifying the predictive factors and applying machine learning models to forecast customer churn, which is crucial for enhancing customer retention strategies in the fast-growing digital payment sector. The research is contextualized within the unique challenges and characteristics of the Iranian market, where mobile payment platforms are gaining significant traction. The authors begin their study by collecting and analyzing a comprehensive dataset from the payment application, which includes user demographic information, transaction history, app usage statistics, and customer service interactions. They apply several data preprocessing techniques to clean and prepare the data for analysis. The research then explores a variety of data mining techniques including logistic regression, decision trees, and support vector machines (SVM) to model and predict churn. The selection of these techniques is justified by their ability to handle large and complex datasets effectively, providing a broad analysis of user behavior and potential churn indicators.

The paper concludes with a comparative analysis of the performance of the various models based on metrics such as accuracy, precision, recall, and F1-score. The results indicate that SVM outperformed other models in predicting customer churn for the payment application. Rezaeian, Haghighi, and Shahrabi discuss the implications of their findings, emphasizing the importance of leveraging advanced data mining techniques to understand customer behavior better and to implement effective retention strategies. They suggest that the insights gained from the study can guide targeted interventions to improve user engagement and reduce churn, which is critical for sustaining growth and competitiveness in the digital payment industry.

Larasati et al. explore the optimization of deep learning models for predicting customer churn. Their study focuses specifically on the application of an Artificial Neural Network (ANN) model, which is known for its ability to learn and model non-linear and complex relationships [16]. The research aims to refine these models to increase their effectiveness in predicting churn, thereby helping businesses in proactive customer retention.

The methodology detailed in the paper involves the use of a deep learning ANN model that has been designed to handle large datasets typically associated with customer interactions and demographics. The authors experimented with various architectures and tuning parameters to optimize the model's performance.

They also employed techniques such as dropout and regularization to prevent overfitting, ensuring that the model generalizes well on unseen data. The optimization process was rigorously tested using a dataset compiled from a telecommunications company, which included features like customer usage patterns, billing information, and personal demographics. The paper concludes with a presentation of the results from the optimized ANN model, showcasing its superiority over traditional machine learning models in terms of accuracy and reliability in predicting customer churn. Larasati et al. discuss the practical implications of their findings, emphasizing how telecom companies can use such optimized deep learning models to identify at-risk customers and implement effective retention strategies. The study highlights the potential of deep learning in transforming the approach to customer retention, making it more data-driven and efficient.

Priya Gopal et al. conducted a comprehensive survey on the methodologies of customer churn prediction in the e-commerce sector using machine learning and data mining techniques. This survey paper provides an overview of various approaches and tools employed to predict customer churn, which is a significant challenge in the rapidly evolving e-commerce industry due to its direct impact on profitability and growth [17]. The authors begin by outlining the key factors contributing to customer churn in e-commerce, such as service quality, price sensitivity, customer satisfaction, and competitive offerings. They then delve into a range of data mining techniques including decision trees, support vector machines, neural networks, and ensemble methods that have been utilized to predict churn. The survey categorizes and compares these techniques based on their effectiveness, scalability, and complexity. Gopal and Mohd Nawi emphasize the role of big data analytics in enhancing the predictive accuracy of these models by enabling the analysis of vast amounts of diverse customer data.

Finally, the paper discusses the challenges and future directions in churn prediction within the e-commerce context. It highlights the importance of integrating more dynamic data sources like social media and web browsing behaviors to better understand customer intentions and improve prediction models. The authors suggest that advancements in artificial intelligence and machine learning could lead to more proactive and personalized churn prevention strategies.

They conclude that ongoing research and adaptation in methodologies are crucial for businesses looking to use technology to reduce churn and enhance customer loyalty in the competitive e-commerce landscape.

Ali et al. investigate the use of supervised learning techniques to predict customer churn in the telecommunications sector. This research focuses on the application of various machine learning models to predict whether customers are likely to leave their service provider, which is a critical challenge in the telecom industry due to its competitive nature and the high cost associated with losing and acquiring customers. The authors examine several supervised learning algorithms, including logistic regression, decision trees, support vector machines (SVM), and neural networks. These models are trained on historical data collected from a telecom company, which includes features like customer demographic details, call usage, billing information, service complaints, and contract details. The goal is to identify patterns and indicators that can predict churn. Each model's performance is evaluated based on metrics such as accuracy, precision, recall, and the area under the receiver operating characteristic curve (AUC), providing a comprehensive analysis of which models perform best in predicting churn.

The paper concludes that among the tested models, the support vector machine (SVM) and neural networks show superior performance in terms of overall prediction accuracy. Ali, Rehman, and Hafeez suggest that the telecom industry can significantly benefit from implementing these models into their customer retention strategies. They also discuss the potential for these techniques to be integrated into real-time analytics tools that can help telecom companies more effectively manage their customer relationships and reduce churn. The study underscores the importance of continuous model tuning and integration of new data to improve the predictive accuracy over time [18].

Mitkees et al. focus on developing a customer churn prediction model using data mining techniques. The study addresses the necessity for effective churn prediction strategies in industries where customer retention is crucial for sustained business success. The authors aim to harness the power of data mining to identify patterns and predictors of customer attrition, allowing businesses to implement targeted interventions.

The research involves applying a range of data mining techniques to a dataset comprising various customer attributes, such as demographic data, service usage patterns, and customer service interactions. The techniques discussed and evaluated include logistic regression, decision trees, and neural networks. The authors carefully preprocess the data, handle missing values, and perform feature selection to optimize the inputs for better model accuracy. The effectiveness of each technique is measured based on criteria such as accuracy, precision, recall, and F1-score.

The paper concludes that decision trees and neural networks, in particular, provide a robust mechanism for predicting customer churn. The results indicate that these models can effectively discriminate between customers likely to churn and those likely to stay, thus providing actionable insights for customer retention strategies. Mitkees, Badr, and ElSeddawy suggest that further research should explore the integration of more complex variables and the application of ensemble techniques to enhance predictive accuracy. The study highlights the potential of data mining techniques in transforming strategic approaches to customer management and retention [19].

Yuwadi Thein Naing et al. review various feature selection methods and techniques applied specifically in customer churn prediction within the telecom industry. The study highlights the critical role of feature selection in enhancing the performance of predictive models by identifying the most relevant variables that influence customer decisions to leave a service provider [20]. This is particularly significant in the telecom sector, where customer retention is a major strategic focus due to intense competition and high acquisition costs. The authors systematically categorize and analyze different feature selection techniques, such as filter methods, wrapper methods, and embedded methods, each distinguished by their approach to integrating with machine learning models. Filter methods evaluate features based on their intrinsic properties without involving models, wrapper methods use a predictive model to assess feature subsets, and embedded methods incorporate feature selection as part of the model training process. The paper evaluates each method's effectiveness in reducing dimensionality and improving model accuracy by comparing them across various studies and datasets within the telecom industry.

The paper concludes with a discussion on the implications of the reviewed feature selection techniques, emphasizing their importance in reducing overfitting, improving model interpretability, and decreasing computational costs.

Naing, Raheem, and Batcha advocate for a more strategic application of feature selection techniques in predictive modeling workflows, suggesting that a well-chosen feature selection method can significantly influence the success of churn prediction models. They also highlight areas for future research, including the exploration of hybrid feature selection methods that combine the strengths of existing approaches to better capture the complexities of customer churn in the telecom industry.

CHAPTER 3

DESIGN OF ANTICIPATING CHURN PREDICTION MODEL

3.1 System architecture

Several elements are typically included in the system architecture for Customer Churn Prediction using Machine Learning to develop, train, and execute a churn prediction model. The first step, data collection, is obtaining and preparing customer data from various sources, such as transaction logs, customer reviews, and demographic data. Following that, the data is modified, cleaned, and prepared for modeling. The second part is feature engineering, which involves choosing and modifying pertinent data properties to raise the prediction model's accuracy. Techniques like feature scaling, feature selection, and dimensionality reduction may be used for this. To efficiently detect and anticipate customer churn inside a firm, the customer churn prediction system architecture consists of several components and phases.

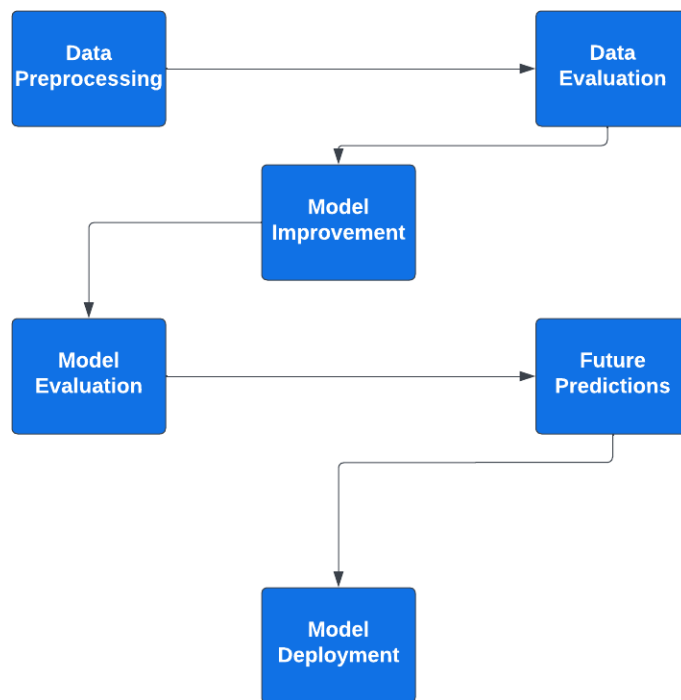


Figure. 3.1 Architecture of the Anticipating Churn Prediction Model

1. Data Preprocessing:

This initial phase involves preparing the raw data for analysis. Tasks include cleaning the data by handling missing values, errors, or outliers; normalizing or scaling the data; encoding categorical variables; feature selection; and sometimes feature engineering, where new features are created from the existing ones.

2. Data Evaluation:

After preprocessing, the data is evaluated to ensure that it's been cleaned and formatted properly. This step may involve exploratory data analysis (EDA) to understand the distributions of the variables, the presence of correlations, or patterns within the data. It ensures that the dataset is ready for modeling.

3. Model Improvement:

This is an iterative process that involves refining the machine learning models. This refinement can include hyperparameter tuning, where the parameters that govern the learning process of the model are adjusted; feature selection to determine which variables are the most predictive; and cross-validation to ensure the model's performance is consistent across different subsets of the data.

4. Model Evaluation:

The model is evaluated using a hold-out set (validation set) or through cross-validation. Performance metrics (like accuracy, precision, recall, F1 score, etc.) are calculated to assess how well the model is predicting. This step may loop back to 'Model Improvement' if the performance is not satisfactory.

5. Future Predictions:

Once a model is evaluated and deemed satisfactory, it can be used to make predictions on new, unseen data. This is the forward-looking aspect of the model, applying it to real-world scenarios to generate insights, inform decisions, or affect outcomes.

6. Model Deployment:

In this final phase, the model is deployed into a production environment where it can provide ongoing predictions. Deployment can involve integrating the model into existing software systems, setting up a service (like a REST API) through which predictions can be made, or

otherwise enabling access to the model's capabilities for end-users.

Each of these steps is crucial for the success of a predictive modeling project, and they represent a cycle that often requires several iterations to optimize the performance and utility of the model. Once the data collection phase is completed after that, the data is kept for later processing and analysis in a central data repository, such a data warehouse or big data platform. The next stage is data preparation, which involves feature engineering, cleaning, and transformation of the collected data.

This step ensures that the data's format is suitable for analysis and modeling. Activities that could be included include handling missing values, data normalization, and feature creation based on domain knowledge. The model creation step of the design comes after data preparation. At this point, statistical or machine learning methods are used to develop prediction models. To find trends and pinpoint the main causes of customer attrition, these models are trained on historical customer data, which includes both non-churned and churned customers. Various techniques, based on the available data and the complexity of the task, can be used, by including logistic regression, decision trees, and neural networks.

Models are evaluated using appropriate performance metrics after development, including area under the curve (AUC), recall, accuracy, and precision.

- The AUC metric refers to the area under the Receiver Operating Characteristic (ROC) curve. This curve plots the true positive rate (sensitivity or recall) against the false positive rate (1 - specificity) at various threshold settings. The AUC measures the ability of a model to discriminate between classes. An AUC of 0.5 suggests no discrimination (equivalent to random chance), while an AUC of 1.0 indicates perfect discrimination. A higher AUC value indicates a better performing model, capable of distinguishing between the positive and negative classes effectively.
- Recall measures the proportion of actual positives (e.g., cases of a disease, churned customers) that are correctly identified by the model. It is calculated as:

$$\text{Recall} = \text{True Positives (TP)} / \text{True Positives (TP)} + \text{False Negatives (FN)}$$

Recall is particularly important in scenarios where missing a positive instance is costly (e.g., failing to diagnose a sick patient).

- Accuracy is one of the most intuitive performance metrics. It measures the proportion

of true results (both true positives and true negatives) in the total data set.

It is calculated as:

$$\text{Accuracy} = \text{True Positives (TP)} + \text{True Negatives (TN)} / \text{Total Population}$$

While accuracy is straightforward, it can be misleading in datasets where the class distribution is imbalanced (e.g., 95% of the data is of one class). In such cases, a model might achieve high accuracy by simply predicting the majority class, but it would be poor at predicting the minority class.

- Precision measures the proportion of positive identifications that were actually correct.

It is calculated as:

$$\text{Precision} = \text{True Positives (TP)} / \text{True Positives (TP)} + \text{False Positives (FP)}$$

Precision is crucial in situations where false positives are a larger concern than false negatives. For example, in email spam detection, a false positive (marking an important email as spam) is more problematic than a false negative (failing to mark a spam email).

- The F1 score is the harmonic mean of precision and recall, giving equal weight to both. It is a way to combine both precision and recall into a single measure that captures both properties. This can be particularly useful when you want to balance the importance of precision and recall. The formula for the F1 score is:

$$\text{F1 Score} = 2 \times (\text{Precision} \times \text{Recall} / \text{Precision} + \text{Recall})$$

Here, if either the precision or the recall is low, the F1 score also will be low. The F1 score's best value is 1 (perfect precision and recall), and the worst is 0.

- The F2 score weights recall higher than precision. It is a measure of a test's accuracy that considers recall to be more important than precision. It is used in scenarios where missing actual positive cases is worse than generating false positives. The formula for the F2 score is:

$$\text{F2 Score} = (1+2^2) \times ((\text{Precision} \times \text{Recall}) / (4 \times \text{Precision}) + \text{Recall})$$

Here, the 2^2 term indicates the weight given to recall over precision. Essentially, F2 score tries to find an effective balance between the precision and the recall, but with more emphasis on recall. In practice, both F1 and F2 scores help to evaluate models where you need a balance between precision and recall but with different emphases depending on the particular costs of false positives versus false negatives. These scores are widely used in classification tasks, including document classification, spam detection, disease screening, and more, where true negatives don't necessarily matter as much as the true

positives and the cost of missing them.

- Each of these metrics offers different insights into the performance of a model, and the importance of each depends on the specific application and the consequences of various types of errors. Often, trade-offs between these metrics must be managed; for example, improving recall might reduce precision and vice versa. Thus, understanding these metrics helps in tuning models according to the specific requirements of the application.

This analysis helps assess the model's predictive accuracy for client attrition. If the model's performance is judged acceptable, it can be used for real-time prediction in a production setting. During the implementation stage, the predictive model needs to be incorporated into the customer relationship management (CRM) platforms or existing business systems. The model uses input data, such as customer characteristics and behavior, to generate churn forecasts for individual customers. Through resource allocation, personalization of customer interactions, and prioritization of retention initiatives, these forecasts can help lower the probability of attrition.

Finally, a feedback loop is included in the system architecture to support the churn prediction model's continuous improvement. By collecting feedback on the forecast accuracy and monitoring the actual churn results, the model may be regularly retrained and adjusted to take into account changing consumer behavior and market dynamics. To sum up, the customer churn prediction system architecture includes data collection, preprocessing, model building, assessment, deployment, and ongoing improvement. It enables businesses to better target customer retention strategies, proactively identify potential department customers and take preventative actions so that the consumers never get any dissatisfaction. Choosing the best machine learning technique for the task at hand is the third component, or model selection. Finding the best approach in terms of accuracy, precision, recall, and F1 score may necessitate comparing several techniques, such as logistic regression, support vector machines, decision trees, and random forests.

The training of the model, which is the fourth component, entails using techniques like cross-validation and hyperparameter adjustment to optimize the model's performance by training the chosen algorithm on the prepared data. To make sure the trained model generalizes well to new data, its performance is assessed on a holdout set of data as the last step. The final

stage, known as "model deployment," entails using the learned model to create predictions by applying it to fresh data. In order to accomplish this, the model might be made available as a microservice or REST API that can be incorporated into already-existing apps. Overall, the Customer Churn Prediction using Machine Learning system architecture has a number of components that require expertise in machine learning algorithms, feature engineering, data pretreatment, and deployment infrastructure. Additionally, the system needs to be scalable, trustworthy, and maintainable in order to handle enormous data volumes and continue making accurate predictions over time.

3.2 Proposed Workflow

Choosing the best machine learning technique for the task at hand is the third component, or model selection and improvement. Finding the best approach in terms of accuracy, precision, recall, and F1 score may necessitate comparing a number of techniques, such as logistic regression, support vector machines, decision trees, and random forests. The training of the model, which is the fourth component, entails using techniques like cross-validation and hyper-parameter adjustment to optimize the model's performance by training the chosen algorithm on the prepared data. To make sure the trained model generalizes well to new data, its performance is assessed on a holdout set of data as the last step which is known as model evaluation. The final stage, known as "model deployment," entails using the learned model to create predictions by applying it to fresh data. In order to accomplish this, the model might be made available as a micro service or REST API that can be incorporated into already-existing apps. Overall, the Customer Churn Prediction using Machine Learning system architecture has a number of components that require expertise in machine learning algorithms, feature engineering, data pretreatment, and deployment infrastructure. Additionally, the system needs to be scalable, trustworthy, and maintainable in order to handle enormous data volumes and continue making accurate predictions over time. Logistic Regression is a statistical method for analyzing a dataset in which there are one or more independent variables that determine an outcome. The outcome is measured with a dichotomous variable (where there are only two possible outcomes).

In our study, LR is used to estimate the probability of churn based on customer features. Kernel Support Vector Machine (SVM) is an extension of the support vector machine that uses a nonlinear mapping to transform the original training data into a higher dimension. In

this space, a linear separator is then constructed. With kernel SVM, we tackle the non-linear relationships between customer attributes and churn, enhancing the model's prediction capability. Random Forest (RF) is an ensemble learning method for classification, regression, and other tasks that operates by constructing a multitude of decision trees at training time. For classification tasks, the output of the random forest is the mode of the classes output by individual trees. RF contributes to the robustness of our predictive model by addressing the variance in predictions. K-Nearest Neighbors (KNN) is a non-parametric method used for classification and regression. In both cases, the input consists of the k closest training examples in the feature space. KNN is included in our approach to incorporate the principle of similarity, where the churn status of similar customers is used to predict the churn likelihood of a new customer. Decision Tree algorithm is a decision support tool that uses a tree-like model of decisions and their possible consequences. It is one way to display an algorithm that only contains conditional control statements. Decision trees help in breaking down a complex decision-making process into more manageable parts, making the model's predictions more interpretable. These models were selected for their diverse approaches to capture various aspects of customer behavior and attributes. By integrating and comparing these models, we aim to harness their collective predictive power for a more accurate and comprehensive churn prediction system.

CHAPTER 4

ANALYSIS OF ANTICIPATING CHURN PREDICTION MODEL

4.1 Existing System

Depending on the company, the sector, and the available data, different approaches may be taken when using machine learning to predict customer attrition. But generally speaking, businesses might use a conventional method of churn prediction that involves manual data analysis, a limited set of variables, and simple statistical models. This method may not be precise or effective, which could result in bad business choices and customer retention tactics.

Some organisations may already be using a basic machine learning model that was trained on sparse and outdated data, and has a predetermined set of characteristics. This approach's inability to account for dynamic shifts in customer behavior and preferences may lead to less accurate forecasts. It's probable that other companies have implemented more advanced machine learning algorithms that utilize a variety of consumer data, including demographics, previous purchases, online behavior, interactions with customer service, and sentiment analysis on social media. These models could use deep learning techniques like decision trees and neural networks to produce accurate and dynamic churn forecasts.

Present customer churn prediction systems typically combine data collection, data analysis, and predictive modeling. Businesses gather relevant information from a range of sources, such as customer interactions, demographic information, and historical transaction history. Databases and data warehouses are used to organize and store this data.

The process of converting and purifying collected data to ensure that it is high-quality and appropriate for analysis is known as data preparation. This step comprises filling in missing values, removing outliers, and normalizing data in order to create a consistent and reliable dataset. When the data is ready, businesses use predictive modeling techniques to build churn prediction models. These models use statistical methods or machine learning algorithms to identify patterns, correlations, and variables that result in customer turnover. Several algorithms, including decision trees, logistic regression, random forests, and neural networks (NNs), are used to train the models using historical customer data.

AUC, accuracy, recall, precision, and other measures are used to test the churn prediction models' performance. This assessment aids in determining how well the model predicts client attrition. Provided the model meets the necessary performance benchmarks, it is integrated into CRM platforms or business systems for real-time churn prediction. In order to assess the model's efficacy within the current system, companies often monitor and compare the churn forecasts with the actual churn outcomes. This feedback loop allows for continuous model development by retraining and updating the churn prediction models based on the latest data and consumer behavior. To provide further insights into high-risk customer categories, retention strategy effectiveness, and churn-related patterns, the existing system might also have features like dashboards or visualisation tools. These visualisations help stakeholders make informed decisions by helping them understand the dynamics of churn and how to reduce it while increasing customer retention.

In general, the present customer churn prediction system includes data collection, preprocessing, predictive modeling, performance evaluation, and ongoing improvement. It can be used by businesses to identify customers who pose a risk, take proactive measures to keep them around, and enhance retention strategies to boost client happiness and loyalty. However, even with advanced machine learning models, certain organizations may have problems with data security, quality, and interpretability that could affect the churn forecasts' dependability and acceptability. Ensuring optimal performance and client satisfaction requires regular evaluation and improvement of the machine learning models used for churn prediction.

4.1 Proposed System

Predictive churning model is a tool for classifying, a system that examines the traits of potential consumers to determine what traits are essential in forecasting turnover rates. Let's imagine we have a dataset with information on ten thousand clients who are taking money out of a bank. These clients' characteristics, including their country of residence, credit score, age, and balance, among others, are described in the data. The outcome of the user's turnover should be predicted by your model. Hence, the target variable will be terminated. The data should be examined with an emphasis as to how various aspects connect to the customer churn status.

To construct many models in search of the optimum fit. Forecasting customer turnover is a problem of binary classification since clients can leave or stay for a predetermined amount of time.

Several elements are typically included in the system architecture for Customer Churn Prediction using Machine Learning in order to develop, train, and execute a churn prediction model as mentioned in Figure 3.1. The first step, data collection, is obtaining and preparing customer data from various sources, such as transaction logs, customer reviews, and demographic data. Following that, the data is modified, cleaned, and prepared for modeling. The second part is feature engineering, which involves choosing and modifying pertinent data properties to raise the prediction model's accuracy. Techniques like feature scaling, feature selection, and dimensionality reduction may be used for this. To efficiently detect and anticipate customer churn inside a firm, the customer churn prediction system architecture consists of several components and phases. Once the data collection phase is completed after that, the data is kept for later processing and analysis in a central data repository, such a data warehouse or big data platform. The data preparation or preprocessing, which involves feature engineering, cleaning, and transformation of the collected data. This step ensures that the data's format is suitable for analysis and modeling. Activities that could be included include handling missing values, data normalization, and feature creation based on domain knowledge. The model creation step of the design comes after data evaluation.

At this point, statistical or machine learning methods are used to develop prediction models. In order to find trends and pinpoint the main causes of customer attrition, these models are trained on historical customer data, which includes both non-churned and churned customers. Various techniques, based on the available data and the complexity of the task, can be used, by including logistic regression, decision trees, and neural networks. Models are evaluated using appropriate performance metrics after development, including area under the curve (AUC), recall, accuracy, and precision. This analysis helps assess the model's predictive accuracy for client attrition. If the model's performance is judged acceptable, it can be used for real-time prediction in a production setting. During the implementation stage, the predictive model needs to be incorporated into the customer relationship management (CRM) platforms or existing business systems.

The model uses input data, such as customer characteristics and behavior, to generate churn forecasts for individual customers.

Through resource allocation, personalization of customer interactions, and prioritization of retention initiatives, these forecasts can help lower the probability of attrition. Last but not least, a feedback loop is included in the system architecture to support the churn prediction model's continuous improvement. By collecting feedback on the forecast accuracy and monitoring the actual churn results, the model may be regularly retrained and adjusted to take into account changing consumer behavior and market dynamics. To sum up, the customer churn prediction system architecture includes data collection, preprocessing, model building, assessment, deployment, and ongoing improvement. It enables businesses to better target customer retention strategies, proactively identify potential department customer and take preventive actions so that the consumers never get any dissatisfaction.

Logistic Regression Classifier

In order to generate an output that falls between 0 and 1, or the likelihood of a given class, the input variables are first transformed using a logistic function in logistic regression. The relationship between the input variables and the event's probability of occurring is displayed using the logistic function. This is the format of the logistic function:

$$P(y=1/x) = 1 / (1 + e^{(-z)}) \quad (4.1)$$

where z is the dot product of the input vector and the model weights, x is the input vector, and $P(y=1/x)$ is the probability of the positive class.

Reducing a loss function: Typically the log loss or cross-entropy loss is how logistic regression is trained. Until the loss is as small as possible, the model weights are adjusted iteratively using an optimization technique such as stochastic gradient descent or the gradient descent method.

The efficient and simple to use algorithm known as logistic regression is straightforward to apply. It can be helpful in situations where the classes are linearly separable and is frequently employed as a baseline model for classification issues. On the other hand, problems with highly correlated features or problems that are non-linearly separable might not yield good results. In such cases, more intricate models such as decision trees, neural networks or

random forests may be more appropriate.

Naive Bayesian Classifier

Based on the Bayes theorem of probability, naive bayes is a probabilistic machine learning technique that uses conditional feature probabilities to divide data into different classes. Naive Bayes is useful in many classification tasks, such as sentiment analysis, recommendation engines, and customer churn prediction, even though it is commonly used in text classification and spam filtering. To find the probability that a data point belongs to a given class, the method multiplies the likelihoods of each feature being assigned a specific class label. As a result, the class with the highest probability is identified as the data point's prediction.

Naive Bayes is a frequently used machine learning algorithm for classification purposes, leveraging Bayes' theorem and operating under the assumption that the features exhibit conditional independence when considering the class label. Despite its simplicity and naïve assumption, it exhibits impressive performance and finds extensive application in a variety of tasks, like spam detection, sentiment analysis, and document classification. The term "naive" is used to describe the algorithm because it makes the assumption that the presence or absence of a specific feature is unrelated to the presence or absence of any other feature, as long as the class label is known.

Naive Bayes counts occurrences in the training data to determine the probabilities of each feature given each class label during the training phase. It analyzes the occurrence frequencies of each class label to determine the initial probabilities for each. The posterior probability of each class label is then ascertained by merging these probabilities and applying Bayes' theorem in light of the observed features. In the prediction stage, Naive Bayes determines the most likely class label for a new instance based on the computed probabilities. To determine the predicted class, it calculates the posterior probabilities for each class label and selects the label with the highest probability.

Naive Bayes is very beneficial. It is efficient in terms of computation and works well with large datasets. It can handle high-dimensional feature spaces and is resilient against unnecessary attributes, as the independence presumption allows it to disregard irrelevant correlations. Naive Bayes is also less prone to overfitting, especially when the training data is limited.

Despite its simplicity, Naive Bayes performs well in many real-world scenarios. But when there are significant dependencies between the features, the feature independence assumption may reduce its usefulness. More complex algorithms might be more appropriate in certain situations. Furthermore, Naive Bayes is susceptible to the existence of uncommon or undiscovered feature combinations in the training set, which may lead to zero probabilities and compromise prediction accuracy.

In conclusion, Naive Bayes is a useful probabilistic algorithm that is applied to classification tasks. It is a well-liked option in many applications due to its effectiveness, ability to handle data with many dimensions, and resilience to unimportant features. Its performance, however, might be constrained in some situations due to its feature independence assumption. Naive Bayes has the advantage of being able to estimate the parameters with a relatively small amount of training data, which is necessary for classification. But in some real-world applications, its assumption of independence might not hold true, and it can be sensitive to unrelated or correlated features.

Kernel SVM

A well-liked machine learning classification algorithm for both linear and non-linear data is Kernel Support Vector Machine (SVM). It operates by identifying the hyperplane within the dataset that maximizes the distance between the two class groups. To facilitate class separation, a kernel function, such as a polynomial function or radial basis function (RBF), is used in conjunction with kernel support vector machines (SVMs) to elevate the data to a higher-dimensional space. Next, the best hyperplane is found using the transformed data. Because it can handle data that is not linearly separable, Kernel Support Vector Machines (SVM) is a potent machine learning algorithm that has gained popularity. Binary classifiers, or SVMs, seek to identify the best hyperplane to divide data points into distinct classes. Nonetheless, the kernel trick is useful when the data cannot be separated linearly.

By transforming the input data into a higher-dimensional feature space and enabling linear separability, kernel SVM expands the capabilities of traditional SVMs. Crucial to the process is the kernel function, which effectively maps data points into the desired space. Radial basis function (RBF), sigmoid, polynomial, and linear are examples of common kernel functions. The kernel trick removes the need for explicit calculations in the higher-dimensional feature space by enabling the SVM algorithm to operate in the original input space. Because of this,

kernel SVM is computationally effective even when dealing with complex data.

One major advantage of kernel SVM is that it can capture complex decision boundaries, which means it can handle non-linear associations in the data with ease. Notably, the RBF kernel is widely used and exhibits remarkable effectiveness in a variety of domains. Kernel SVMs excel in averting overfitting since they prioritize widening the margin between support vectors instead of striving to precisely accommodate each training point. Support vectors, crucial for identifying the ideal hyperplane, are the data points closest to the decision boundary.

While kernel SVMs offer several advantages, they come with certain factors to bear in mind. The selection of a suitable kernel function and the fine-tuning of its parameters can pose a challenge, necessitating meticulous experimentation. Furthermore, when dealing with extensive datasets, kernel SVMs can become computationally intensive, particularly as the training complexity escalates with the growth of support vectors. To sum it up, the kernel SVM stands as a versatile algorithm that capitalizes on the kernel trick to effectively address non-linear data. Its capacity to capture intricate decision boundaries renders it a valuable asset in various machine learning endeavors, encompassing both classification and regression. Nonetheless, the proper choice of Adjusting the kernel and parameters is still essential to achieving optimal performance. Kernel Support Vector Machines (SVM) are useful when the data is non-linear and has complex class boundaries. They are widely used in many different fields, such as bioinformatics, text classification, and image classification.

K- Nearest Neighbor (KNN)

The k-nearest-neighbors (KNN) algorithm is a categorization technique that operates by identifying the k information points that are closest to the point under classification in the feature space. The algorithm then assigns the class with the highest frequency among the k closest neighbors to the point that needs to be classified. The k value is a hyperparameter in KNN that needs to be set before the algorithm can run. A smoother decision boundary that is less sensitive to data noise will be produced by a higher value of k than a smaller one that is more flexible and more sensitive to data noise.

KNN is a simple and effective method that can be applied to problems with both regression and classification.

Because it requires calculating the distances between each data point and every other data point in the dataset, it can be computationally expensive, particularly when working with large datasets. KNN also requires strict normalization of the feature values to avoid features with larger scales dominating the estimated distances.

KNN has some things to think about, though. Appropriate values for the number of neighbors and the distance metric are critical for optimal performance, as it can be sensitive to these choices. In conclusion, K-Nearest Neighbors (KNN) is an easy-to-understand algorithm that bases its prediction on how close training examples are to one another. It's a well-liked option for many machine learning tasks because of its adaptability, resilience to outliers, and simplicity of implementation. However, careful parameter selection and potential scalability issues should be taken into account while using KNN in practical situations.

Support Vector Machine with Radial basis function kernel

Support Vector Machine (SVM) is a supervised machine learning technique that can be used for regression or classification issues. The Radial Basis Function (RBF) kernel is one of the SVM kernels that is most frequently used. It converts the input data into a higher dimensional space and enables the separation of data points using a hyperplane. The RBF kernel is defined by a distance metric that measures how similar two data points are to one another. It is a recommended option for SVM due to its ability to handle non-linearly separable data and simulate complex decision boundaries.

Finding the hyperplane that splits the data points into the appropriate classes and has the biggest margin is the goal. The margin is the separation between the closest data points from each class and the hyperplane. SVM aims to maximize this margin in order to generalize to previously unknown data in an efficient manner. The RBF kernel in SVM, which establishes the distance between data points in the higher dimensional space, may support more complex decision boundaries.

These models need to be worked on and we'll do so using the the given steps:

- **Search for Parameters:** We will select the values and parameters that we wish to consider in each of our models. The GridSearchCV will be run with the optimal model parameters selected.

- **Best Models Fit:** We train the system using the train dataset after determining the best estimator.
- **Performance Evaluation:** We will assess the models that fared the best after being trained on our training dataset using our test set.

4.2 Implementation of Anticipating Customer Churn Prediction Model

1. Data Visualization

Significant findings from your data may support the growth of the business. The issue is that conclusions cannot always be made by merely glancing at the data. When you visually analyze your information, patterns, links, and other startling discoveries that might not be apparent otherwise become apparent[11]. By using data visualization to make your data come to life and reveal its hidden meanings, you can hone your storytelling abilities. Data visualization enables users to quickly and effectively create compelling business insights through graphical displays, dynamic reports, live data visualizations, and numerous other illustrations.

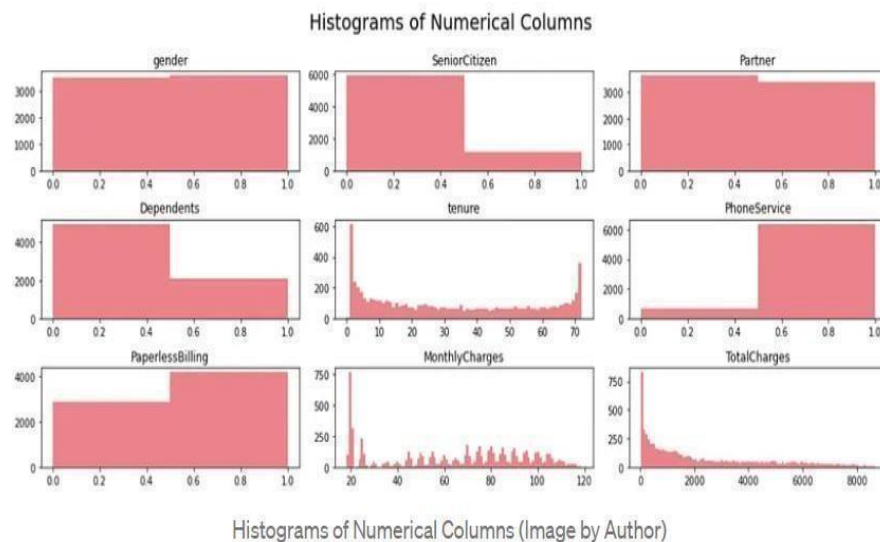


Figure 4.1. Histogram of numerical data

Exploring and visualizing the data set by doing distribution of independent variables to better understand the patterns in the data and to potentially form some hypothesis is shown in Figure 4.1.

Through dynamic reports, graphical presentations, live data visualizations, and other visuals, data visualization enables users to quickly and effectively provide appealing business insights. The gender distribution of the dataset shows that the proportion of male and female clients is about equal. There are exactly equal numbers of males and women in our dataset. We can draw a bunch of conclusions based on the histograms represented in Figure 4.1. Data visualization is the process of presenting data in a visual format, such as charts, graphs, or maps, for understanding, analysis, and communication of information. It transforms complex datasets into visual representations that are more accessible and intuitive for humans to comprehend.

Through data visualization, patterns, trends, and relationships within the data can be easily identified and interpreted. It enables people to visually examine the distributions, variations, and correlations between various variables to explore and gain insights from the data. Making data-driven decisions and identifying trends are made simpler when data is presented visually. Different kinds of visualizations can be used, based on the type of data and the intended use. Bar charts, line charts, scatter plots, pie charts, histograms, heatmaps, and geographical maps are examples of visualizations that are frequently used. Every kind of visualization has a distinct function in depicting various facets of the data, like contrasting values, exhibiting temporal trends, showcasing category composition, or presenting spatial patterns.

Data visualization plays a crucial role in data analysis and decision-making across numerous domains, including business, finance, healthcare, marketing, and research. It enables stakeholders to gain a holistic view of complex datasets and effectively communicate insights to a wide range of audiences. Moreover, interactive data visualizations allow users to interact with the data and customize the visual representations based on their needs. They can zoom in, filter, and manipulate the data to explore specific aspects or drill down into details.

In summary, data visualization is a powerful tool for transforming data into meaningful and actionable insights. It simplifies complex information, uncovers patterns, and facilitates effective communication of data-driven findings. By leveraging visual representations, individuals can make informed decisions, drive innovation, and gain a deeper understanding of the underlying data.

- The gender distribution of the dataset indicates that the proportion of male and female

clients is about equal. There are the same number of men and women in our dataset. Younger clients make up the majority of the dataset's customers.

- It appears that few users have dependents, even though 50% of users share their plan with a partner.
- The company boasts a loyal customer base that has been with them for an average of over 70 months, in addition to a large number of recent clients.
- Seventy-five percent of customers prefer transactions without paper, and the majority of them seem to need access to a cell phone. Invoice fees per user each month vary between eighteen dollars to one hundred and eighteen dollars, with a large majority of consumers falling into the twenty dollars bracket.

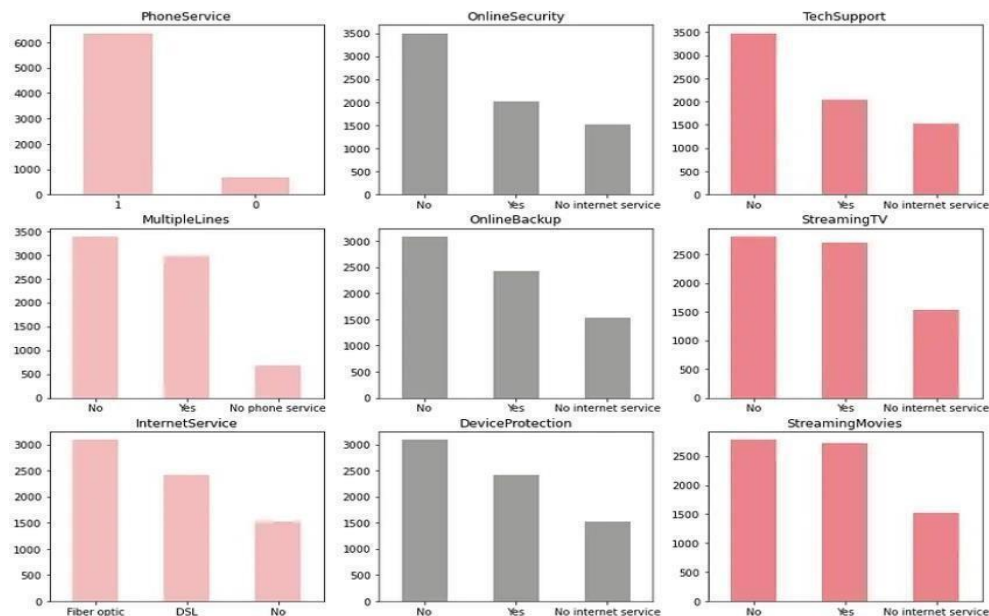


Figure 4.2. Distribution of Label Encoded Categorical Variables

From the plots of Figure 4.2, we learn several things:

- Almost all consumers have access to cell phone service, and nearly half have multiple lines of service.
- Over 50% of internet users watch movies and TV series online, and 3/4 of users select DSL and fiber-optic lines for their internet connectivity.
- The internet backup features, tech assistance, and safety precautions have only been utilized by a very small percentage of customers.

The plots teach us various things, including:

- Customers who are leaving are typically older than those who stay.
- Regarding tenure or the median credit score, there is no difference between clients who are maintained and those who are lost.
- Most of the customers who quit the company seem to still have a substantial amount in their bank accounts.
- The number of items or the expected wage do not seem to have an impact on customer churn.

We need to undertake some feature engineering before we search for a model to forecast customer turnover.

To create the most recent characteristics that better describe our clients, several of the dataset's attributes are combined. Although, as we have previously noted, credit scores do not affect churning, they typically increase with time and subsequently with age, we will be developing a New feature is accounted to this so that there will be no loss of customers.

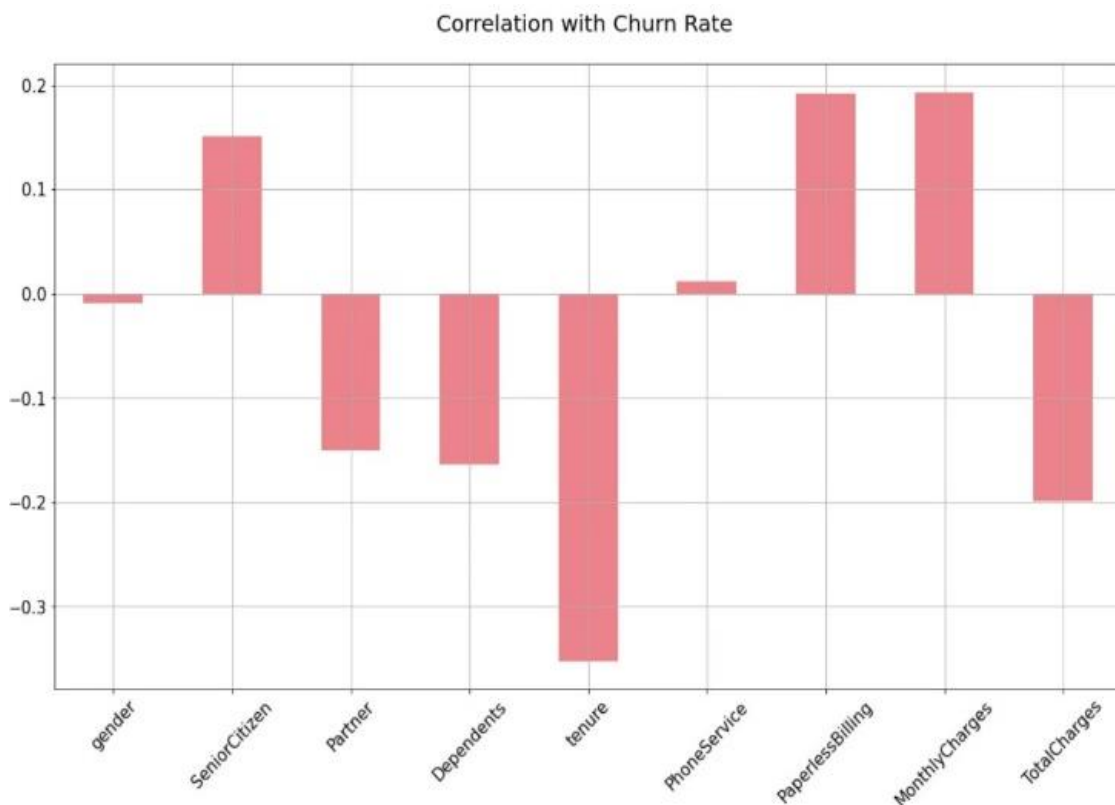


Figure 4.3 Correlation with churn rate

Positive and negative correlations with churn rate are shown graphically in Figure 4.3.

Correlation matrix helps us to discover the bivariate relationship between independent variables in a dataset. Which is basically a statistical technique used to evaluate the relationship between two variables in a data set. The churn rate increases with monthly charges and age. In contrast Partner, Dependents and Tenure seem to be negatively related to churn.

Table 4.1 Comparative Analysis of various classification techniques

Algorithm	ROC AUC Mean	ROC AUC STD	Accuracy Mean	Accuracy STD
Logistic Regression	84.12	1.65	74.6	1.26
SVC	83.64	1.68	79.98	1.08
Gaussian NB	81.82	1.79	68.99	1.46
Random Forest	81.72	2.02	78.47	1.57
Kernel SVM	79.66	2.12	79.85	1.08
KNN	77.04	2.38	75.77	1.09
Decision Tree Classifier	65.44	1.67	72.75	1.45

Classification Accuracy is a widely used metric for evaluating baseline algorithms. It measures the proportion of right predictions out of the total number of predictions. Nevertheless, it is not the optimal measure when we encounter a class imbalance problem. Therefore, let us arrange the outcomes according to the 'Mean AUC' metric, which represents the model's capacity to differentiate between positive and negative categories. Table 4.1 presents a comparative analysis of different classification techniques.

2. Exploratory Data Analysis

An important step in the data analysis process is called exploratory data analysis (EDA), which focuses on understanding the data at a basic level before using more sophisticated methods. To extract insights and spot patterns, trends, and outliers, the dataset is systematically explored, visualized, and summarized.

Data analysts use a range of methods and visualizations during EDA to find significant features and connections in the data. They use box plots, density plots, or histograms to

analyze the variable distribution in order to determine the overall shape, central tendency, and dispersion of the data. To evaluate the connections between various variables and spot possible dependencies or associations, they can also make use of scatter plots or correlation matrices.

EDA also entails determining how to handle data that contains outliers, missing values, or inconsistent data. Using the proper methods, analysts can impute missing values or eliminate outliers that could have a major effect on the analysis. Preprocessing and cleaning procedures are carried out to ensure that the data is suitable for further analysis. Additionally, EDA makes it possible to find pertinent subgroups or subsets within the data. Analysts can compare and contrast patterns and relationships within each subgroup by segmenting the dataset based on distinct categorical variables. This allows for more focused and insightful analyses.

Analysts can create hypotheses and research questions for additional study by using exploratory data analysis. The variables that have the greatest impact on explaining the desired outcome can be identified, which aids in directing feature selection in later modeling endeavors. Additionally, EDA aids in identifying possible biases, anomalies, or problems with the quality of the data that could affect the accuracy of later analyses.

In general, the initial stage of data exploration and comprehension is exploratory data analysis. It reveals patterns and relationships, sheds light on the characteristics of the dataset, and influences decisions made later on in the modeling or analysis process. Through the process of visualizing and summarizing data, analysts are able to derive valuable insights from complex datasets, generate hypotheses, and make well-informed decisions. Key findings of EDA performed on this model

- The dataset contains no inaccurate or missing data values.
- While Partner, Dependents, and Tenure have the strongest negative correlations with the goal qualities, Monthly Charges and Age have the strongest positive correlations.
- The dataset is unbalanced since many consumers have motion.
- There is a multicollinear relationship between monthly charges and total charges. The removal of Total Charges has resulted in a significant decrease in the VIF values.
- The majority of the dataset's customers are younger.
- The bulk of the company's clientele is comprised primarily of new customers (those

under a year old), with a devoted following of clients older than 70 months.

- Most users seem to have phone service; each user pays between \$18 and \$118 per month. A large portion of customers with monthly subscriptions have a good chance of doing so as well if they have opted to pay with online checks.

3. Classification Models

Because of the number of accurate forecasts made relative to all predictions, classification precision is one of the most popular classification assessment indicators used to evaluate baseline techniques [5]. However, it is not the most useful statistic when there are problems with class disparities. The data will be classified using the "Mean AUC" score, which measures how well the model's predictions can distinguish between both favorable and unfavorable classes [4].

Based on the dataset's highest mean AUC Scores, the first cycle of foundation algorithms for classification showed that the logistic regression model and SVC outperformed the other five models. When we look at the Accuracy scores graphically in Figure 5, we can see that logistic regression performs well when compared to the other methods. Machine learning relies heavily on classification models, which are widely used to predict class labels or categorical outcomes based on input features. Numerous well-liked classification models exist, each with unique traits, benefits, and use cases.

One common method of classification is logistic regression, which uses a logistic function to determine the relationship between input features and the probability of falling into a particular class.

Challenges of binary and multi-class classification are skillfully handled by this linear model. Logistic regression is a very interpretable and computationally efficient method that works well with a wide range of dataset sizes. Decision trees are flexible classification models that use a tree-like structure to generate options. Each inner node in the tree represents a feature, and the branches represent possible feature values. Decision trees have the benefit of being easy to understand and depict, and they can handle both numerical and category attributes. However, they are prone to overfitting, especially as the tree becomes large and complex.

An ensemble learning method called the Random Forest algorithm combines multiple decision trees to create predictions. Its strategy for reducing the overfitting issue with

Decision Trees involves adding randomness via bootstrapping and randomly selecting features. Random Forest produces robust and accurate results even in the presence of missing or noisy data. It is also adept at handling large-scale datasets.

Robust classifiers called Support Vector Machines (SVMs) look for the optimal hyperplane for class differentiation. SVMs reduce the chance of overfitting by optimizing the margin, which improves class separation. When a kernel function is applied to transform data into a higher-dimensional feature space, they can be applied to both linearly and non-linearly separable data. SVMs perform well on moderately sized datasets, but they can be computationally demanding on larger datasets.

Based on the premise that features are conditionally independent given the class label, the Naive Bayes model is a classification technique that draws its foundation from Bayes' theorem. Efficient computation and training are greatly aided by this feature independence. This model performs well in managing large datasets and traversing feature spaces with many dimensions. However, because of the assumption of independence, it might not be able to capture complex feature dependencies. Neural Networks, and particularly Deep Learning models, have become increasingly popular for classification tasks in recent years. These models can understand complex data relationships because they are made up of interconnected nodes, or neurons, layered upon layers. Although Deep Learning models require large amounts of data to train and are computationally demanding, they have demonstrated state-of-the-art performance in a variety of domains, including text and image classification.

These examples only show a small number of classification models, each with unique benefits and limitations. The specific problem at hand, the characteristics of the data, and the desired trade-offs between interpretability, accuracy, and computing speed all play a role in selecting the best model. Attaining the best classification results requires understanding the nuances of each model and experimenting with different strategies.

CHAPTER 5

RESULTS AND DISCUSSIONS

Classification Accuracy is a widely used metric for evaluating baseline algorithms. It measures the proportion of right predictions out of the total number of predictions. Nevertheless, it is not the optimal measure when we encounter a class imbalance problem. Therefore, let us arrange the outcomes according to the 'Mean AUC' metric, which represents the model's capacity to differentiate between positive and negative categories. Table 5.1 presents a comparative analysis of different classification techniques.

Table 5.1 Comparison of Baseline Classification Algorithms - 1st Iteration

Algorithm	ROC AUC Mean	ROC AUC STD	Accuracy Mean	Accuracy STD
Logistic Regression	84.12	1.65	74.6	1.26
SVC	83.64	1.68	79.98	1.08
Gaussian NB	81.82	1.79	68.99	1.46
Random Forest	81.72	2.02	78.47	1.57
Kernel SVM	79.66	2.12	79.85	1.08
KNN	77.04	2.38	75.77	1.09
Decision Tree Classifier	65.44	1.67	72.75	1.45

Table 5.2 Comparison of Baseline Classification Algorithms – 2nd Iteration

Model	Accuracy	Precision	Recall	F1 Score	F2 Score
Logistic Regression	0.803407	0.652038	0.55615	0.600289	0.573003
SVM (Linear)	0.803407	0.650155	0.561497	0.602582	0.57724
Kernel SVM	0.791341	0.637931	0.494652	0.557229	0.517917
Random Forest	0.779276	0.6171	0.44385	0.51633	0.470255
K-Nearest Neighbors	0.76863	0.570175	0.52139	0.544693	0.530468
Decision Tree	0.739532	0.508997	0.529412	0.519004	0.525199
Naive Byes	0.703336	0.467359	0.842246	0.601145	0.725806

The second iteration of comparison of base line classification of algorithms is shown in Table 5.2. In the subsequent phase of evaluating baseline classification methods, the optimized parameters for KNN and Random Forest models will be employed. Additionally, it is understood that false negatives incur higher costs compared to false positives in a churn scenario. Therefore, we will utilize precision, recall, and F2 scores as the optimal metrics for selecting the model. In the subsequent phase of evaluating baseline classification methods, the optimized parameters for KNN and Random Forest models will be employed. Additionally, it is understood that false negatives incur higher costs compared to false positives in a churn scenario. Therefore, we will utilize precision, recall, and F2 scores as the optimal metrics for selecting the model. All things considered, the models function well, and logistic regression proved to be the most helpful in this instance. As a result, efforts to improve this model have been concentrated, and the accuracy has increased. The confusion matrix shown in Figure 5.1 represents the ultimate outcome.

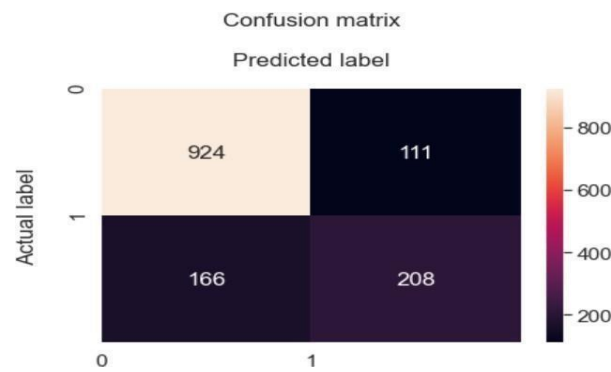


Figure. 5.1 Confusion Matrix

The Confusion matrix shows that we have 208+924 correct predictions and 166+111 incorrect ones. At 80% accuracy, our model exhibits the characteristics of a credible model.

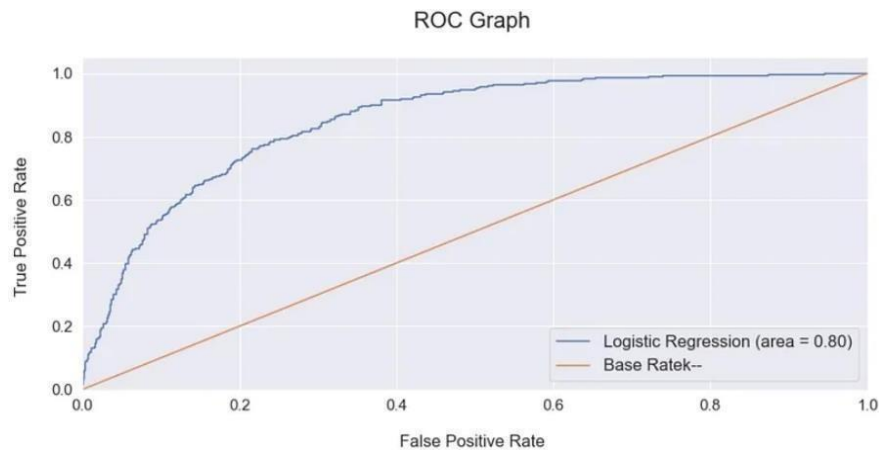


Figure. 5.2 ROC Graph

Reassessing the system using the Receiver Operating Characteristic graph makes sense. In accordance with the AUC Mean score Figure 5.2 ROC graph illustrates how well a model can distinguish between different categories. A machine learning model aims to minimize the Base Rate, represented by the orange line on the ROC curve of a random classifier. The enhanced Logistic Regression model had a higher area under the curve score, as can be seen in the graph above. Because of the number of accurate forecasts made relative to all predictions, classification precision is one of the most popular classification assessment indicators used to evaluate baseline techniques [5]. However, it is not the most useful statistic when there are problems with class disparities. Thus, the data will be categorized using the "Mean AUC" score, which measures how well the model's predictions can distinguish between both favorable and adverse classes [4].

The comparison of the algorithms and their corresponding accuracy is shown in Table 5.1. Based on the dataset's highest mean AUC Scores, the first cycle of foundation algorithms for classification showed that the logistic regression model and SVC outperformed the other five models. When the Accuracy scores are compared graphically in Figure 5.2, we can observe that logistic regression performs well in terms of accuracy when compared to other methods. The ranks are divided into 10 quantiles, effectively dividing the entire dataset into 10 groups of equal size. Then labels are assigned to these quantiles ranging from 10 to 1, where 10 would be assigned to the highest propensity scores and 1 to the lowest. The ultimate goal of employing a propensity score is to mitigate selection bias by establishing equivalence between groups determined by these factors. By employing this approach, the validity of comparisons between the treatment and control groups in terms of outcomes is enhanced, thereby closely resembling the observations that would be made in a randomized controlled trial. Using propensity score we divide the customers into two categories.

- *Category – Highest Risk for Churn*

Table 5.3 Determining churn rate with prediction of propensity score – high risk.

Index	customerID	Churn	predictions	propensity_to_churn(%)	Ranking
5532	9300-AGZNL I	1	1	85.99	1
5173	5567-WSELE I	1	1	84.87	1
7010	1415-YFWLT I	1	1	83.25	1
6507	2446-BEGGB I	1	1	83.22	1
5985	5277-ZLOOR I	1	1	83.08	1
5636	8884-ADFDN I	1	1	82.44	1
6667	6630-UJZMY I	1	1	82.38	1
6501	2865-TCHJW I	1	1	82.22	1
2367	4750-ZRXIU I	1	1	82.2	1
1018	2660-EMUB I	1	1	81.8	1

- *Category – Lowest Risk for Churn*

Table 5.4 Determining churn rate with prediction of propensity score – low risk.

<i>Index</i>	<i>customerID</i>	<i>Churn</i>	<i>predictions</i>	<i>propensity_to_churn(%)</i>	<i>Ranking</i>
5532	4957 - SREEC	0	0	0.43	10
5173	1403 - GYAFD	0	0	0.44	10
7010	1052 - QJIBV	0	0	0.46	10
6507	8590 - YFFQO	0	0	0.47	10
5985	5787 - KXGIY	0	0	0.48	10
5636	0464 - WJTKO	0	0	0.49	10
6667	7876 - DNYAP	0	0	0.5	10
6501	3642 - BYHDO	0	0	0.5	10
2367	8229 - BUJHX	0	0	0.51	10
1018	3678- MNGZX	0	0	0.52	10

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENT

6.1 Conclusion

The logistic regression model predicts that a monthly contract, an optical fiber internet connection, online payments, the lack of a payment security guarantee, and inadequate technical support will all contribute to an increase in the turnover rate. In contrast, the model predicts a negative correlation with churn if a customer has a one-year contract, subscribes to online security, or chooses to pay with postal checks. Many churn prediction models currently in use ignore the larger market or industry context in favor of treating each customer as an isolated entity. Our model has successfully addressed this problem, and in the future, it may be possible to add additional contextual data to churn prediction models, such as competitive behavior or economic trends. The majority of churn prediction models in use today rely on structured data, such as demographic and transaction histories. In the future, it may be possible to incorporate unstructured data into churn prediction models, such as social media posts or customer reviews, to provide a more thorough examination of consumer behavior. This has the potential to be improved upon in the future.

In the current highly competitive telecommunications industry, churn prediction is a crucial concern for customer relationship management (CRM). Its purpose is to retain valuable customers by identifying groups of customers with similar characteristics and offering them competitive offers and services. Classification accuracy is a quantitative measure used to assess the efficiency of classification methods. This measure is often used for assessing basic classification methods. However, when there are significant differences in social classes, this statistic becomes less useful. It becomes evident that logistic regression outperforms the other approaches in terms of accuracy. Overall, the models perform effectively, and logistic regression demonstrated the highest utility in this particular case. As a result, efforts to develop this model have been concentrated, and the accuracy has increased. In this work, a customer churn-over model for data analytics is presented and then assessed using standard evaluation metrics. The results obtained show that our suggested churn model performs better thanks to the application of machine learning methods. With an ROC AUC mean of 88%, Logistic regression produced a better result. After analyzing the dataset, we determined

the primary factors that contribute to customer churn. Then it was conducted cluster profiling based on the level of risk associated with each element. Ultimately, we furnished decision-makers of telecom corporations with explicit instructions on customer retention. In the future, the study might be expanded to investigate the evolving behavioral patterns of churn consumers with the application of Artificial Intelligence approaches for predictive and trend analysis. So, we conclude that we made use of a customer churn dataset from Kaggle to build a machine learning classifier that predicts the propensity of any customer to churn in months to come with a reasonable accuracy score of 80% to 84%.

6.2 Future Enhancement

The project can be enhanced by incorporating advanced techniques such as ensemble learning, deep learning, and reinforcement learning to improve prediction accuracy and robustness. Additionally, leveraging big data technologies for handling large-scale datasets, implementing real-time monitoring systems to detect churn signals promptly, and integrating feedback loops to continuously improve model performance based on new data and evolving customer behaviors will be crucial. Furthermore, ensuring ethical considerations such as fairness, transparency, and privacy preservation throughout the development and deployment phases will foster trust and acceptance of the predictive models among stakeholders and customers, ultimately leading to more effective churn prevention strategies and increased customer retention rates.

REFERENCES

- [1] Nyoman Mahayasa Adiputra, Paweena Wanchai, “Customer Churn Prediction Using Weight Average Ensemble Machine Learning Model,” 2023 20th International Joint Conference on Computer Science and Software Engineering (JCSSE).
- [2] Diao Azzam, Manar Hamed, Nora Kasiem, Yomna Eid, Walaa Medhat, “Customer Churn Prediction Using Apriori Algorithm and Ensemble Learning,” 2023 5th Novel Intelligent and Leading Emerging Sciences Conference (NILES).
- [3] Yunjie Lin, Mu Shengdong, Gu jijian, Nadia Nedjah, “Intelligent prediction using customer churn with a Fused attentional Deep learning model,” 2022 Multidisciplinary Digital Publishing Institute (MDPI),China.
- [4] Parth Pulkundwar, Krishna Rudani, Omkar Rane, Chintan Shah, Shyamal Virnodkar, “A Comparison of Machine Learning Algorithms for Customer Churn Prediction” 2023 6th International Conference on Advances in Science and Technology (ICAST).
- [5] Eunjo Lee, Boram Kim, Sungwook, Kang, Byungsoo Kang, Yoonjae Jang, and Huy Kang Kim,”Profit Optimizing Churn Prediction For Long Term Loyal Customers in Online Games,” 2020 IEEE Transactions on Games.
- [6] Xin Hu, Yanfei Yang, Lanhua Chen , Siru Zhu, “Research on a customer Churn Combination Prediction Model Based on Decision Tree and Neural Network,” 2020 IEEE 5th International Conference on Cloud Computing and Big Data Analytics (ICCCBDA).
- [7] Harish A S, Malathy C, “Evaluative study of cluster based customer churn prediction against conventional RFM based churn model”, 2023 Second International Conference on Electrical, Electronics, Information and Communication Technologies (ICEEICT).
- [8] Kiran Deep Singh, Prabh Deep Singh, Ankit Bansal, Gaganpreet Kaur, Vikas Khullar, Vikas Tripathi, “Exploratory Data Analysis and Customer Churn Prediction for the Telecommunication Industry,” 2023 3rd International Conference on Advances in Computing, Communication, Embedded and Secure Systems (ACCESS).
- [9] Mustafa Büyükkeçeci, Mehmet Cudi Okur, “A Data Mining Application in Customer Churn Prediction,” 2022 IEEE Conference.
- [10] S. Venkatesh, M. Jeyakarthic, “An Optimal Genetic Algorithm with Support Vector Machine for Cloud Based Customer Churn Prediction,” 2020 IEEE Conference.
- [11] Aishwarya H M, Bindhiya T, S Tanisha, Soundarya B, C Christlin Shanuja, “Customer Churn Prediction Using Synthetic Minority Oversampling Technique,” 2023 4th International Conference on Communication, Computing and Industry 6.0 (C216).
- [12] Abhishek Gaur, Ratnesh Dubey, “Predicting Customer Churn Prediction In Telecom Sector

Using Various Machine Learning Techniques,” 2018 International Conference on Advanced Computation and Telecommunication (ICACAT).

- [13] Mohamed Galal, Sherine Rady, Mostafa Aref, "Enhancing Customer Churn Prediction in Digital Banking using Ensemble Modeling,” 2022 4th Novel Intelligent and Leading Emerging Sciences Conference (NILES).
- [14] Fulu Chen, Xiaowei Wei, Saisai Yu, Pengfei Ma, Shuqing He, "Customer Churn Prediction based on Stacking Model,” 2023 4th International Conference on Computer Vision, Image and Deep Learning (CVIDL).
- [15] O. Rezaeian, S. S. Haghighi and J. Shahrabi, "Customer Churn Prediction Using Data Mining Techniques for an Iranian Payment Application," 2021 12th International Conference on Information and Knowledge Technology (IKT), Babol, Iran, Islamic Republic, 2021.
- [16] A. Larasati, D. Ramadhanti, Y. W. Chen, A. Muid, "Optimizing Deep Learning ANN Model to Predict Customer Churn," 2021 7th International Conference on Electronics Engineering (ICEEIE),Malang, Indonesia, 2021.
- [17] Priya Gopal, Nazri Bin MohdNawi, “A Survey on Customer Churn Prediction using Machine Learning and data mining Techniques in E-commerce,” 2021 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE).
- [18] M. Ali, A. U. Rehman and S. Hafeez, "Prediction of Churning Behavior of Customers in Telecom Sector Using Supervised Learning Techniques," 2018 IEEE 3rdInternational Conference on Computing,
- [19] I. M. M. Mitkees, S. M. Badr and A. I. B. ElSeddawy, "Customer churn prediction model using data mining techniques," 2017 13th International Computer Engineering Conference (ICENCO), Cairo, Egypt.
- [20] Yuwadi Thein Naing, Mafas Raheem, Nowshath K Batcha, “Feature Selection for Customer Churn Prediction: A Review on the Methods & Techniques applied in the Telecom Industry,” 2022 IEEE International Conference on Distributed Computing and Electrical Circuits and Electronics (ICDCECE).

APPENDIX 1

```
# Step 0: Restarting the session and Clearing all temporary variables-----

try:
    from IPython import get_ipython
    get_ipython().magic('clear')
    get_ipython().magic('reset -f')
except:
    pass
```

Import required libraries:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from scipy.stats import norm, skew
from scipy import stats
import statsmodels.api as sm

from sklearn.impute import SimpleImputer
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

from sklearn import svm, tree, linear_model, neighbors
from sklearn import naive_bayes, ensemble, discriminant_analysis, gaussian_process
from sklearn.neighbors import KNeighborsClassifier
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from xgboost import XGBClassifier
```

```

from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import confusion_matrix, accuracy_score
from sklearn.metrics import f1_score, precision_score, recall_score, fbeta_score
from statsmodels.stats.outliers_influence import variance_inflation_factor
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import GridSearchCV

from sklearn.model_selection import ShuffleSplit
from sklearn.model_selection import KFold

from sklearn import feature_selection
from sklearn import model_selection
from sklearn import metrics

from sklearn.metrics import classification_report, precision_recall_curve
from sklearn.metrics import auc, roc_auc_score, roc_curve

[ ] from sklearn.metrics import classification_report, precision_recall_curve
    from sklearn.metrics import auc, roc_auc_score, roc_curve
    from sklearn.metrics import make_scorer, recall_score, log_loss
    from sklearn.metrics import average_precision_score

import seaborn as sn
from matplotlib import pyplot
import matplotlib.pyplot as plt
import matplotlib.pylab as pylab
import matplotlib
%matplotlib inline
color = sn.color_palette()
import matplotlib.ticker as mtick
from IPython.display import display
pd.options.display.max_columns = None
from pandas.plotting import scatter_matrix
from sklearn.metrics import roc_curve

import random
import os
import re
import sys
import timeit

```

```

from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import confusion_matrix, accuracy_score
from sklearn.metrics import f1_score, precision_score, recall_score, fbeta_score
from statsmodels.stats.outliers_influence import variance_inflation_factor
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import GridSearchCV

from sklearn.model_selection import ShuffleSplit
from sklearn.model_selection import KFold

from sklearn import feature_selection
from sklearn import model_selection
from sklearn import metrics

from sklearn.metrics import classification_report, precision_recall_curve
from sklearn.metrics import auc, roc_auc_score, roc_curve

```

```

[ ] from sklearn.metrics import classification_report, precision_recall_curve
    from sklearn.metrics import auc, roc_auc_score, roc_curve
    from sklearn.metrics import make_scorer, recall_score, log_loss
    from sklearn.metrics import average_precision_score

import seaborn as sn
from matplotlib import pyplot
import matplotlib.pyplot as plt
import matplotlib.pylab as pylab
import matplotlib
%matplotlib inline
color = sn.color_palette()
import matplotlib.ticker as mtick
from IPython.display import display
pd.options.display.max_columns = None
from pandas.plotting import scatter_matrix
from sklearn.metrics import roc_curve

import random
import os
import re
import sys
import timeit

```

```
import random
import os
import re
import sys
import timeit
import string
import time
from datetime import datetime
from time import time
from dateutil.parser import parse
import joblib
```

Evaluate the Dataset

ps.chdir(r"C:/Users/srees/Desktop/Blogs and Projects/Upcoming Blogs and Projects/2. Propensity Scoring Models/3. Predict Customer Churn/")

```
[ ] dataset = pd.read_csv('1.Input/customer_churn_data.csv')
```

```
[ ] dataset.head()
```

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	OnlineBackup	D
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	DSL	No	Yes	
1	5575-GNVDE	Male	0	No	No	34	Yes	No	DSL	Yes	No	
2	3668-QPYBK	Male	0	No	No	2	Yes	No	DSL	Yes	Yes	
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	DSL	Yes	No	
4	9237-HQITU	Female	0	No	No	2	Yes	No	Fiber optic	No	No	

```
[ ] dataset.columns
```

```
Index(['customerID', 'gender', 'SeniorCitizen', 'Partner', 'Dependents',
      'tenure', 'PhoneService', 'MultipleLines', 'InternetService',
      'OnlineSecurity', 'OnlineBackup', 'DeviceProtection', 'TechSupport',
      'StreamingTV', 'StreamingMovies', 'Contract', 'PaperlessBilling',
      'PaymentMethod', 'MonthlyCharges', 'TotalCharges', 'Churn'],
      dtype='object')
```

```
[ ] dataset.describe()
```

 `dataset.describe()`



	SeniorCitizen	tenure	MonthlyCharges
count	7043.000000	7043.000000	7043.000000
mean	0.162147	32.371149	64.761692
std	0.368612	24.559481	30.090047
min	0.000000	0.000000	18.250000
25%	0.000000	9.000000	35.500000
50%	0.000000	29.000000	70.350000
75%	0.000000	55.000000	89.850000
max	1.000000	72.000000	118.750000

Finding Missing Values by Re-Evaluating Columns

To validate the column datatypes for missing values, you can use the `info()` method in pandas to display information about the DataFrame, including the column datatypes and the number of non-null values in each column.

The output will display the datatype for each column and the number of non-null values in each column. If there are missing values in a column, the number of non-null values will be less than the total number of rows in the DataFrame.

If you want to check the number of missing values in each column, you can use the `isnull()` method in pandas to create a Boolean DataFrame that indicates which values are missing, and then use the `sum()` method to count the number of missing values in each column.

```
[ ] dataset.dtypes
```

```
customerID      object
gender          object
SeniorCitizen    int64
Partner         object
Dependents      object
tenure          int64
PhoneService    object
MultipleLines   object
InternetService object
OnlineSecurity  object
OnlineBackup    object
DeviceProtection object
TechSupport     object
StreamingTV     object
StreamingMovies object
Contract        object
PaperlessBilling object
PaymentMethod   object
MonthlyCharges  float64
TotalCharges    object
Churn           object
dtype: object
```

```
[ ] #look for missing
dataset.columns.to_series().groupby(dataset.dtypes).groups

{dtype('int64'): Index(['SeniorCitizen', 'tenure'], dtype='object'),
 dtype('float64'): Index(['MonthlyCharges'], dtype='object'),
 dtype('O'): Index(['customerID', 'gender', 'Partner', 'Dependents', 'PhoneService',
                    'MultipleLines', 'InternetService', 'OnlineSecurity', 'OnlineBackup',
                    'DeviceProtection', 'TechSupport', 'StreamingTV', 'StreamingMovies',
                    'Contract', 'PaperlessBilling', 'PaymentMethod', 'TotalCharges',
                    'Churn'],
                    dtype='object')}
```



```
[ ] dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7043 entries, 0 to 7042
Data columns (total 21 columns):
#   Column                Non-Null Count  Dtype
---  -
0   customerID            7043 non-null   object
1   gender                 7043 non-null   object
2   SeniorCitizen          7043 non-null   int64
3   Partner                7043 non-null   object
4   Dependents             7043 non-null   object
5   tenure                 7043 non-null   int64
6   PhoneService           7043 non-null   object
7   MultipleLines          7043 non-null   object
8   InternetService        7043 non-null   object
9   OnlineSecurity         7043 non-null   object
10  OnlineBackup           7043 non-null   object
11  DeviceProtection       7043 non-null   object
12  TechSupport            7043 non-null   object
13  StreamingTV            7043 non-null   object
14  StreamingMovies        7043 non-null   object
15  Contract               7043 non-null   object
16  PaperlessBilling       7043 non-null   object
17  PaymentMethod          7043 non-null   object
18  MonthlyCharges         7043 non-null   float64
19  TotalCharges           7043 non-null   object
```

```
[ ] from sklearn.metrics import classification_report, precision_recall_curve
    from sklearn.metrics import auc, roc_auc_score, roc_curve
    from sklearn.metrics import make_scorer, recall_score, log_loss
    from sklearn.metrics import average_precision_score

import seaborn as sn
from matplotlib import pyplot
import matplotlib.pyplot as plt
import matplotlib.pylab as pylab
import matplotlib
%matplotlib inline
color = sn.color_palette()
import matplotlib.ticker as mtick
from IPython.display import display
pd.options.display.max_columns = None
from pandas.plotting import scatter_matrix
from sklearn.metrics import roc_curve

import random
import os
import re
import sys
import timeit
```



```
dataset.isna().any()
```

customerID	False
gender	False
SeniorCitizen	False
Partner	False
Dependents	False
tenure	False
PhoneService	False
MultipleLines	False
InternetService	False
OnlineSecurity	False
OnlineBackup	False
DeviceProtection	False
TechSupport	False
StreamingTV	False
StreamingMovies	False
Contract	False
PaperlessBilling	False
PaymentMethod	False
MonthlyCharges	False
TotalCharges	False
Churn	False
dtype:	bool



```
dataset.isna().any()
```



customerID	False
gender	False
SeniorCitizen	False
Partner	False
Dependents	False
tenure	False
PhoneService	False
MultipleLines	False
InternetService	False
OnlineSecurity	False
OnlineBackup	False
DeviceProtection	False
TechSupport	False
StreamingTV	False
StreamingMovies	False
Contract	False
PaperlessBilling	False
PaymentMethod	False
MonthlyCharges	False
TotalCharges	False
Churn	False
dtype:	bool


```
[ ] #Unique values in each categorical variable

dataset["PaymentMethod"].nunique()

dataset["PaymentMethod"].unique()

array(['Electronic check', 'Mailed check', 'Bank transfer (automatic)',
      'Credit card (automatic)'], dtype=object)
```

```
[ ] dataset["Contract"].nunique()

dataset["Contract"].unique()

array(['Month-to-month', 'One year', 'Two year'], dtype=object)
```

```
[ ] #Target Variable Distribution
```

```
dataset["Churn"].value_counts()
```

```
No      5174
Yes     1869
Name: Churn, dtype: int64
```

```
[ ] #Clean the Dataset
```

```
dataset['TotalCharges'] = pd.to_numeric(dataset['TotalCharges'],errors='coerce')

dataset['TotalCharges'] = dataset['TotalCharges'].astype("float")
```

```
[ ] dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7043 entries, 0 to 7042
Data columns (total 21 columns):
#   Column                Non-Null Count  Dtype
---  -
0   customerID            7043 non-null   object
1   gender                 7043 non-null   object
2   SeniorCitizen          7043 non-null   int64
3   Partner                7043 non-null   object
4   Dependents             7043 non-null   object
5   tenure                 7043 non-null   int64
6   PhoneService           7043 non-null   object
7   MultipleLines          7043 non-null   object
8   InternetService        7043 non-null   object
9   OnlineSecurity         7043 non-null   object
10  OnlineBackup           7043 non-null   object
11  DeviceProtection       7043 non-null   object
12  TechSupport            7043 non-null   object
13  StreamingTV            7043 non-null   object
14  StreamingMovies        7043 non-null   object
15  Contract               7043 non-null   object
16  PaperlessBilling       7043 non-null   object
17  PaymentMethod          7043 non-null   object
18  MonthlyCharges         7043 non-null   float64
19  TotalCharges           7032 non-null   float64
20  Churn                  7043 non-null   object
dtypes: float64(2), int64(2), object(17)
memory usage: 1.1+ MB
```

```
[ ] dataset.isna().any()
```

```
customerID      False
gender           False
SeniorCitizen    False
Partner          False
Dependents       False
tenure           False
PhoneService     False
MultipleLines    False
InternetService  False
OnlineSecurity   False
OnlineBackup     False
DeviceProtection False
TechSupport      False
StreamingTV      False
StreamingMovies  False
Contract         False
PaperlessBilling False
PaymentMethod    False
MonthlyCharges   False
TotalCharges     True
Churn            False
dtype: bool
```

```
[ ] #average and fill missing values of each columns
```

```
na_cols = dataset.isna().any()

na_cols = na_cols[na_cols == True].reset_index()

na_cols = na_cols["index"].tolist()

for col in dataset.columns[1:]:
    if col in na_cols:
        if dataset[col].dtype != 'object':
            dataset[col] = dataset[col].fillna(dataset[col].mean()).round(0)
```

```
[ ] #Revalidate:
```

```
dataset.isna().any()
```

```
customerID      False
gender          False
SeniorCitizen   False
Partner         False
Dependents      False
tenure          False
PhoneService    False
MultipleLines   False
InternetService False
OnlineSecurity  False
OnlineBackup    False
DeviceProtection False
TechSupport     False
StreamingTV     False
StreamingMovies False
Contract        False
PaperlessBilling False
PaymentMethod   False
MonthlyCharges  False
TotalCharges    False
Churn           False
dtype: bool
```

```
[ ] #label Encode Binary data
#Create a label encoder object
le = LabelEncoder()

#2 or less unique values
le_count = 0
for col in dataset.columns[1:]:
    if dataset[col].dtype == 'object':
        if len(list(dataset[col].unique())) <= 2:
            le.fit(dataset[col])
            dataset[col] = le.transform(dataset[col])
            le_count += 1
print('{} columns were label encoded.'.format(le_count))
```

```
6 columns were label encoded.
```

```
[ ]
#Exploratory Data Analysis

dataset2 = dataset[['gender', 'SeniorCitizen', 'Partner', 'Dependents',
                    'tenure', 'PhoneService', 'PaperlessBilling',
                    'MonthlyCharges', 'TotalCharges']]

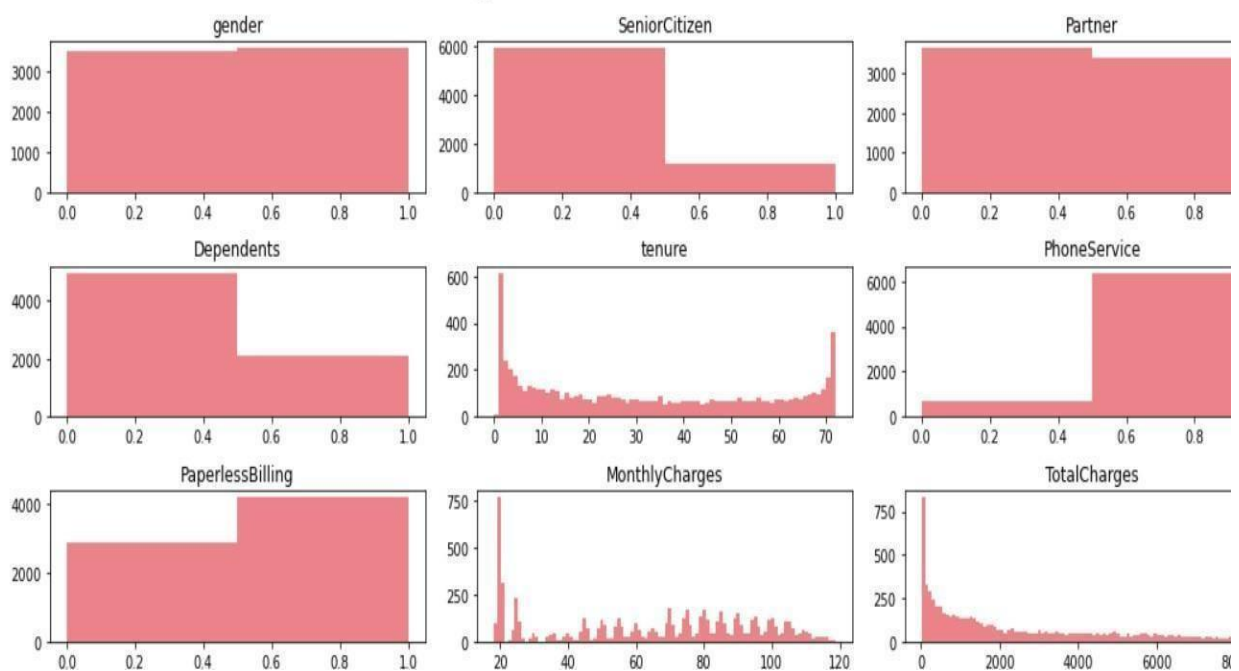
#Histogram:

fig = plt.figure(figsize=(15, 12))
plt.suptitle('Histograms of Numerical Columns\n',horizontalalignment="center",fontstyle = "normal", fontsize = 24, fontfamily = "sans-
for i in range(dataset2.shape[1]):
    plt.subplot(6, 3, i + 1)
    f = plt.gca()
    f.set_title(dataset2.columns.values[i])

    vals = np.size(dataset2.iloc[:, i].unique())
    if vals >= 100:
        vals = 100

    plt.hist(dataset2.iloc[:, i], bins=vals, color = '#ec838a')
plt.tight_layout(rect=[0, 0.03, 1, 0.95])
```

Histograms of Numerical Columns



```
#Analyze distribution of Key Categorical Variables

#(1) Distribution of Contract Type

contract_split = dataset[["customerID", "Contract"]]
sectors = contract_split.groupby("Contract")
contract_split = pd.DataFrame(sectors["customerID"].count())
contract_split.rename(columns={'customerID': 'No. of customers'}, inplace=True)

ax = contract_split[["No. of customers"]].plot.bar(title = 'Customers by Contract Type', legend = True, table = False, grid = False,
subplots = False, figsize = (12, 7), color = '#ec838a', fontsize = 15, stacked=False)

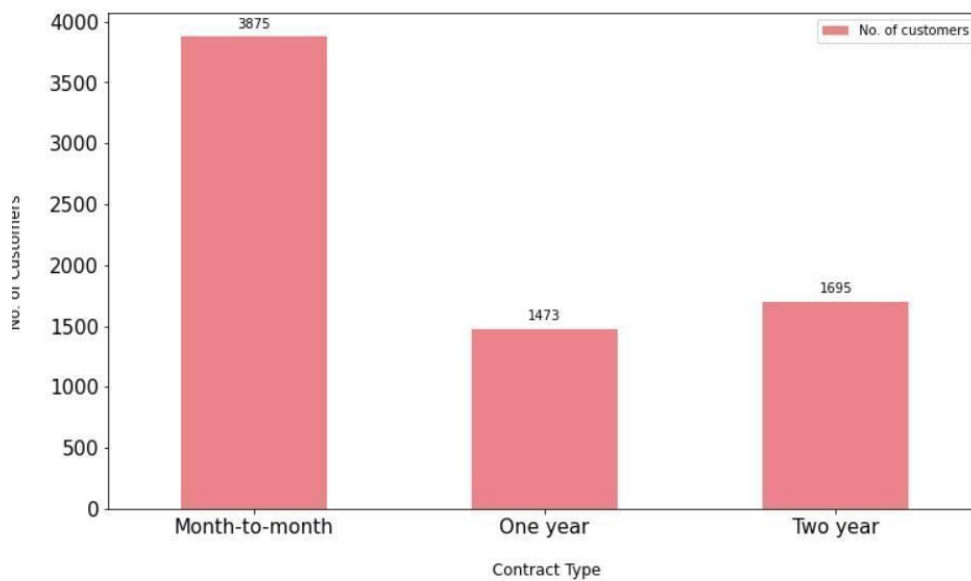
plt.ylabel('No. of Customers\n',horizontalalignment="center",fontstyle = "normal", fontsize = "large", fontfamily = "sans-serif")
plt.xlabel('\n Contract Type',horizontalalignment="center",fontstyle = "normal", fontsize = "large", fontfamily = "sans-serif")
plt.title('Customers by Contract Type \n',horizontalalignment="center", fontstyle = "normal", fontsize = "22", fontfamily = "sans-serif")
plt.legend(loc='top right', fontsize = "medium")
plt.xticks(rotation=0, horizontalalignment="center")
plt.yticks(rotation=0, horizontalalignment="right")
```

```
x_labels = np.array(contract_split[["No. of customers"]])

def add_value_labels(ax, spacing=5):
    for rect in ax.patches:
        y_value = rect.get_height()
        x_value = rect.get_x() + rect.get_width() / 2
        space = spacing
        va = 'bottom'
        if y_value < 0:
            space *= -1
            va = 'top'
        label = "{:.0f}".format(y_value)
        ax.annotate(
            label,
            (x_value, y_value),
            xytext=(0, space),
            textcoords="offset points",
            ha='center',
            va=va)
add_value_labels(ax)
```

```
<ipython-input-21-5789d8875306>:17: MatplotlibDeprecationWarning: Unrecognized location 'top right'. Falling back on 'best'; valid locations
best
upper right
upper left
lower left
lower right
right
center left
center right
lower center
upper center
center
This will raise an exception in 3.3.
plt.legend(loc='top right', fontsize = "medium")
```

Customers by Contract Type



#(2) Distribution of Payment Method Type

```
payment_method_split = dataset[["customerID", "PaymentMethod"]]
sectors = payment_method_split.groupby("PaymentMethod")
payment_method_split = pd.DataFrame(sectors["customerID"].count())
payment_method_split.rename(columns={'customerID': 'No. of customers'}, inplace=True)

ax = payment_method_split[["No. of customers"]].plot.bar(title = 'Customers by Payment Method', legend = True, table = False, grid = False,
subplots = False, figsize = (15, 10), color = '#ec838a', fontsize = 15, stacked = False)

plt.ylabel('No. of Customers\n', horizontalalignment = 'center', fontstyle = "normal", fontsize = "large", fontfamily = "sans-serif")
plt.xlabel('\n Contract Type', horizontalalignment = "center", fontstyle = "normal", fontsize = "large", fontfamily = "sans-serif")
plt.title('Customers by Payment Method\n', horizontalalignment = "center", fontstyle = "normal", fontsize = "22", fontfamily = "sans-serif")
plt.legend(loc = 'top right', fontsize = "medium")
plt.xticks(rotation = 0, horizontalalignment = "center")
plt.yticks(rotation = 0, horizontalalignment = "right")

x_labels = np.array(payment_method_split[["No. of customers"]])

def add_value_labels(ax, spacing=5):
    for rect in ax.patches:
        y_value = rect.get_height()
        x_value = rect.get_x() + rect.get_width() / 2
        space = spacing
        va = 'bottom'
        if y_value < 0:
            space *= -1
            va = 'top'
```

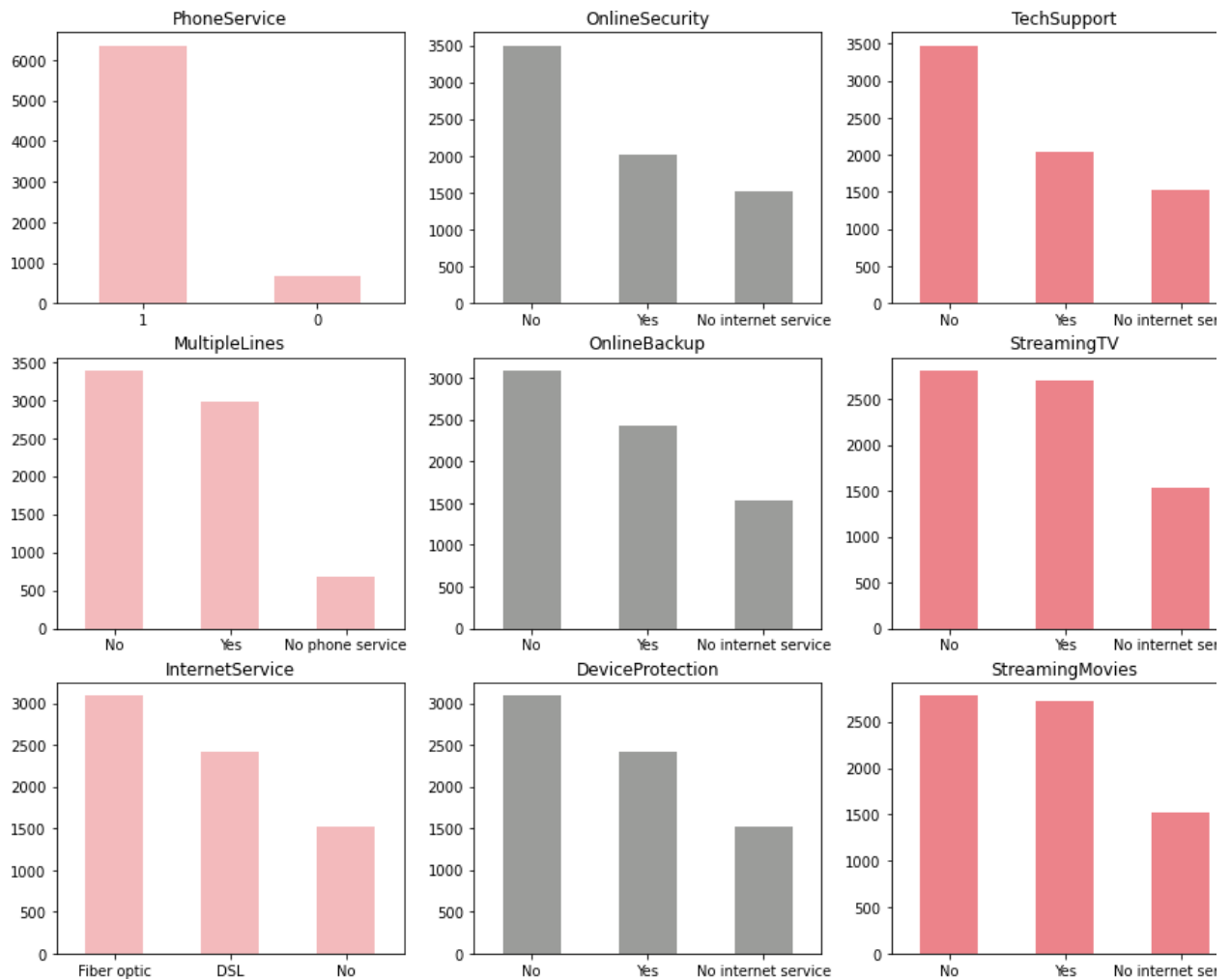
#(3) Distribution of various Label Encoded Categorical Variables

```
services = ['PhoneService', 'MultipleLines', 'InternetService', 'OnlineSecurity',
'OnlineBackup', 'DeviceProtection', 'TechSupport', 'StreamingTV', 'StreamingMovies']

fig, axes = plt.subplots(nrows = 3, ncols = 3, figsize = (15, 12))
for i, item in enumerate(services):
    if i < 3:
        ax = dataset[item].value_counts().plot(kind = 'bar', ax=axes[i,0], rot = 0, color = '#f3babc' )

    elif i >= 3 and i < 6:
        ax = dataset[item].value_counts().plot(kind = 'bar', ax=axes[i-3,1], rot = 0, color = '#9b9c9a')

    elif i < 9:
        ax = dataset[item].value_counts().plot(kind = 'bar', ax=axes[i-6,2], rot = 0, color = '#ec838a')
    ax.set_title(item)
```

```
#Analyze Churn Rate by Categorical variables:

#(1) Overall Churn Rate

import matplotlib.ticker as mtick
churn_rate = dataset[["Churn", "customerID"]]
churn_rate["churn_label"] = pd.Series(np.where((churn_rate["Churn"] == 0), "No", "Yes"))
sectors = churn_rate.groupby("churn_label")
churn_rate = pd.DataFrame(sectors["customerID"].count())
churn_rate["Churn Rate"] = (churn_rate["customerID"] / sum(churn_rate["customerID"])) * 100
ax = churn_rate[["Churn Rate"]].plot.bar(title = 'Overall Churn Rate', legend = True, table = False, grid = False, subplots = False,
figsize = (12, 7), color = '#ec838a', fontsize = 15, stacked = False, ylim = (0, 100))

plt.ylabel('Proportion of Customers', horizontalalignment="center", fontstyle = "normal", fontsize = "large", fontfamily = "sans-serif")
plt.xlabel('Churn', horizontalalignment="center", fontstyle = "normal", fontsize = "large", fontfamily = "sans-serif")
plt.title('Overall Churn Rate \n', horizontalalignment="center", fontstyle = "normal", fontsize = "22", fontfamily = "sans-serif")
plt.legend(loc='top right', fontsize = "medium")
plt.xticks(rotation=0, horizontalalignment="center")
plt.yticks(rotation=0, horizontalalignment="right")
ax.yaxis.set_major_formatter(mtick.PercentFormatter())
x_labels = np.array(churn_rate[["customerID"]])

def add_value_labels(ax, spacing=5):
    for rect in ax.patches:
        y_value = rect.get_height()
        x_value = rect.get_x() + rect.get_width() / 2
        space = spacing
        va = 'bottom'
```

```

def add_value_labels(ax, spacing=5):
    for rect in ax.patches:
        y_value = rect.get_height()
        x_value = rect.get_x() + rect.get_width() / 2
        space = spacing
        va = 'bottom'
        if y_value < 0:
            space *= -1
            va = 'top'

        label = "{:.1f}%".format(y_value)
        ax.annotate(
            label,
            (x_value, y_value),
            xytext=(0, space),
            textcoords="offset points",
            ha='center',
            va=va)
    add_value_labels(ax)
    ax.autoscale(enable=False, axis='both', tight=False)

```

#(2) Churn Rate by Contract Type

```

import matplotlib.ticker as mtick

contract_churn = dataset.groupby(['Contract', 'Churn']).size().unstack()

contract_churn.rename(columns={0: 'No', 1: 'Yes'}, inplace=True)

colors = ['#ec838a', '#9b9c9a']

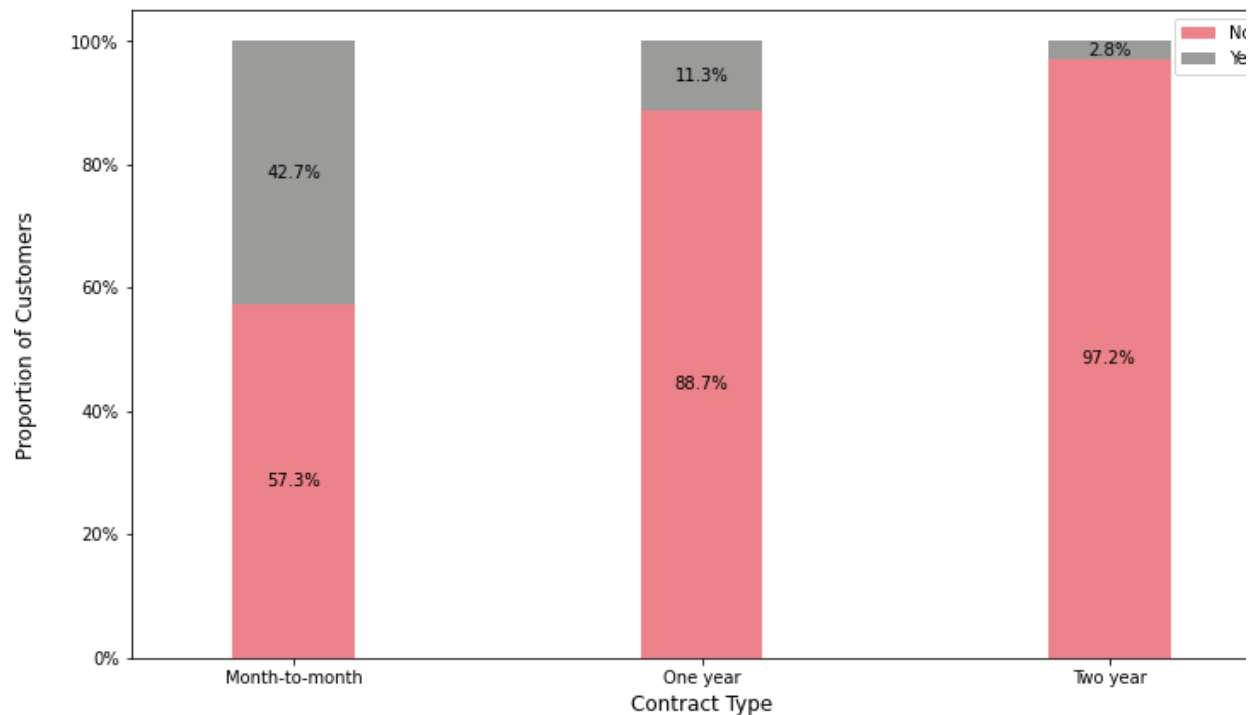
ax = (contract_churn.T*100.0 / contract_churn.T.sum()).T.plot(kind='bar',
                                                                width = 0.3,
                                                                stacked = True,
                                                                rot = 0,
                                                                figsize = (12,7),
                                                                color = colors)

```



```
plt.legend(loc='top right', fontsize = "medium")
```

Churn Rate by Contract type



```
#Find positive and negative correlations with the Response Variable
```

```
dataset2 = dataset[['SeniorCitizen', 'Partner', 'Dependents',
                    'tenure', 'PhoneService', 'PaperlessBilling',
                    'MonthlyCharges', 'TotalCharges']]
```

```
correlations = dataset2.corrwith(dataset.Churn)
```

```
correlations = correlations[correlations!=1]
```

```
positive_correlations = correlations[correlations > 0].sort_values(ascending = False)
```

```
negative_correlations = correlations[correlations < 0].sort_values(ascending = False)
```

```
print('Most Positive Correlations: \n', positive_correlations)
```

```
print('\nMost Negative Correlations: \n', negative_correlations)
```

Most Positive Correlations:

MonthlyCharges	0.193356
PaperlessBilling	0.191825
SeniorCitizen	0.150889
PhoneService	0.011942

dtype: float64

Most Negative Correlations:

Partner	-0.150448
Dependents	-0.164221
TotalCharges	-0.199426
tenure	-0.352229

dtype: float64

```
[ ]
# Plot positive & negative correlation with Response Variable

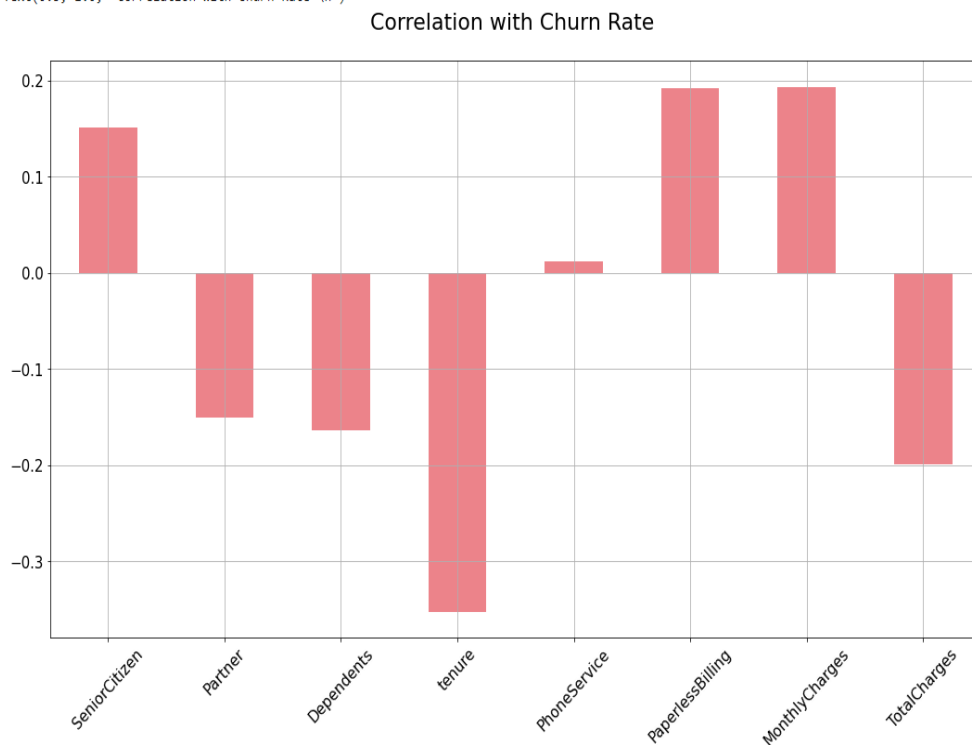
correlations = dataset2.corrwith(dataset.Churn)
correlations = correlations[correlations!=1]

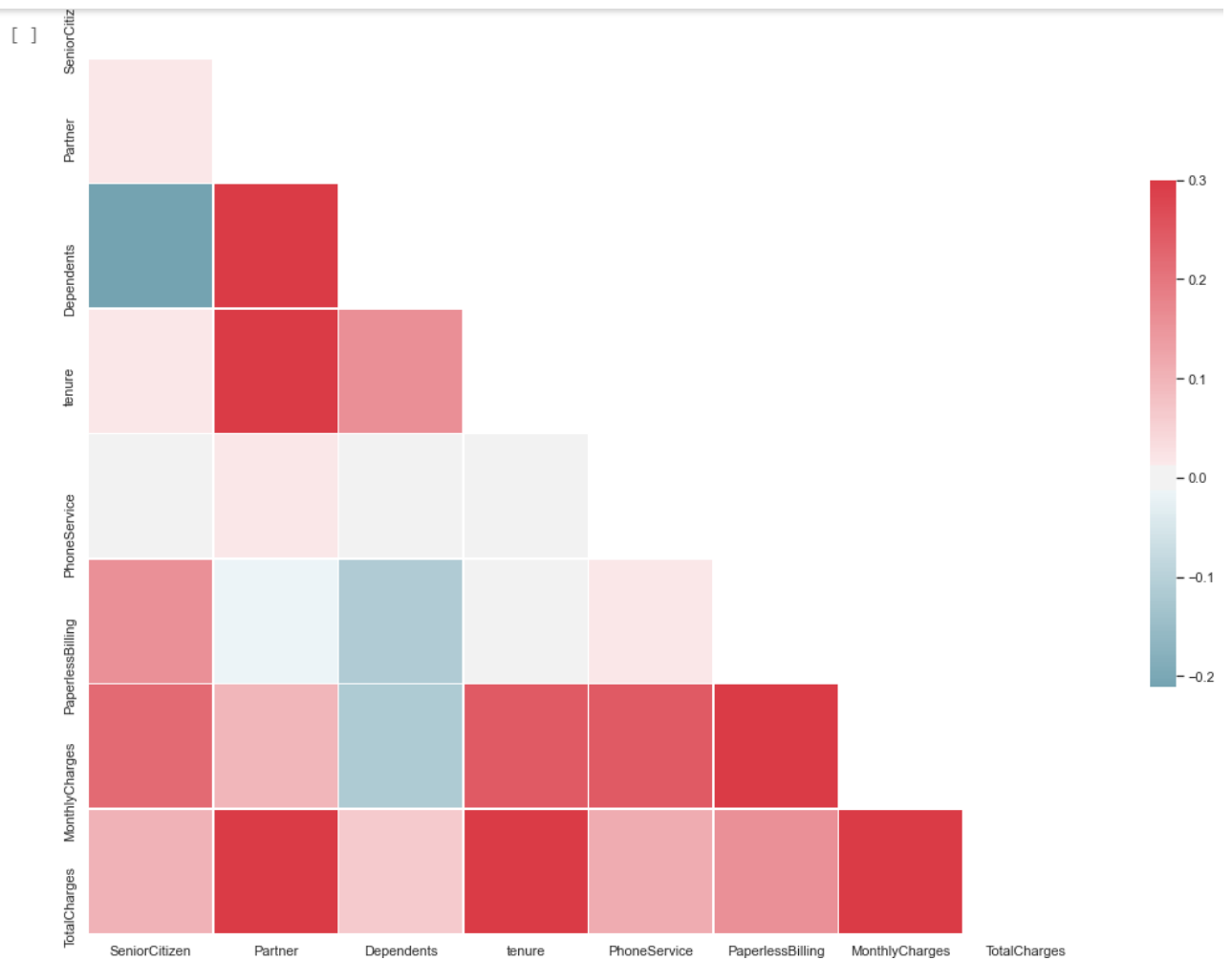
correlations.plot.bar(figsize = (18, 10), fontsize = 15, color = '#ec838a', rot = 45, grid = True)

plt.title('Correlation with Churn Rate \n',horizontalalignment="center", fontstyle = "normal", fontsize = "22", fontfamily = "sans-seri
```

```
Text(0.5, 1.0, 'Correlation with Churn Rate \n')
```

```
Text(0.5, 1.0, 'Correlation with Churn Rate \n')
```





```
#Check Multicollinearity using VIF

def calc_vif(X):

    # Calculating VIF
    vif = pd.DataFrame()
    vif["variables"] = X.columns
    vif["VIF"] = [variance_inflation_factor(X.values, i) for i in range(X.shape[1])]

    return(vif)

dataset2 = dataset[['gender', 'SeniorCitizen', 'Partner', 'Dependents', 'tenure', 'PhoneService', 'PaperlessBilling', 'MonthlyCharges', 'TotalCharges']]
calc_vif(dataset2)
```

	variables	VIF
0	gender	1.921286
1	SeniorCitizen	1.327766
2	Partner	2.815272
3	Dependents	1.921208
4	tenure	10.549667
5	PhoneService	7.976386
6	PaperlessBilling	2.814160
7	MonthlyCharges	13.988649
8	TotalCharges	12.570269

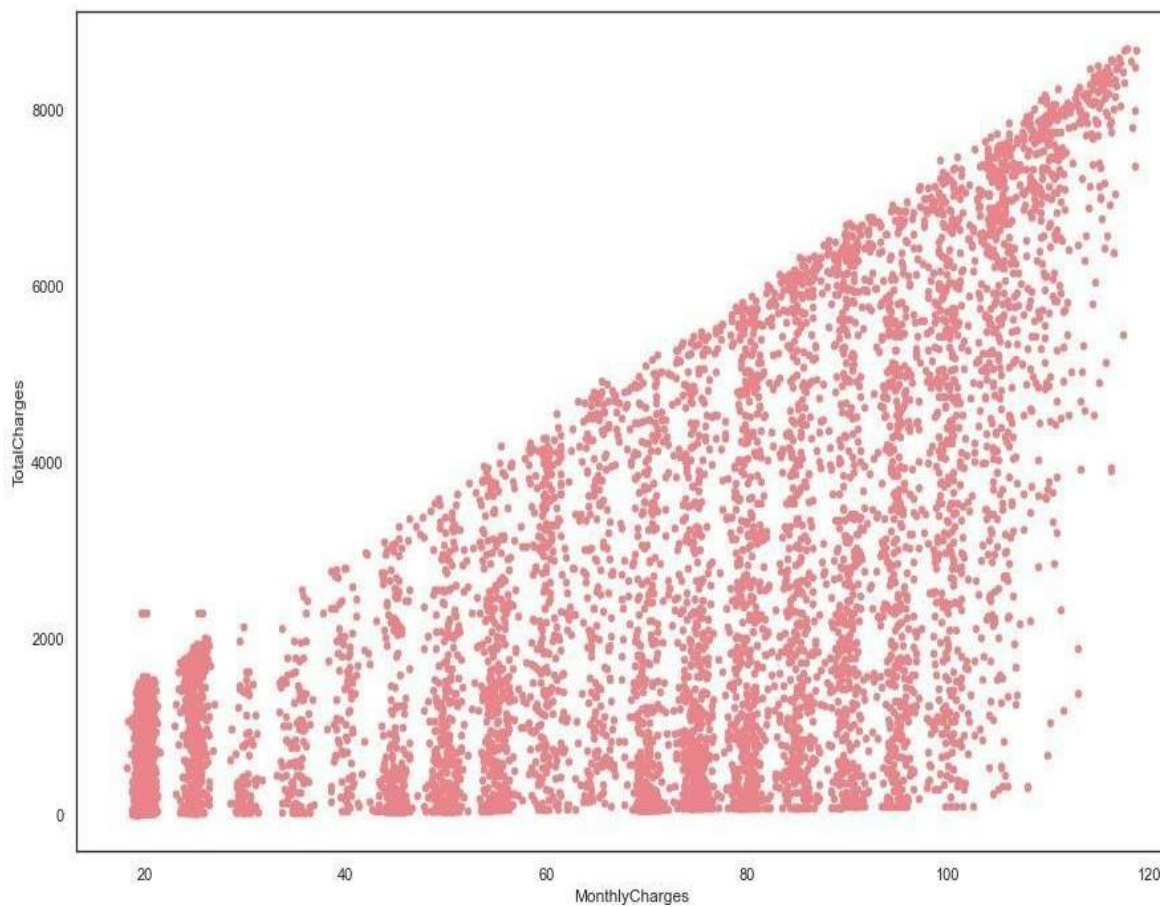
```
#Total Charges seem to be colinear with Monthly Charges.

#check colinearity:

dataset2[['MonthlyCharges', 'TotalCharges']].plot.scatter(figsize = (15, 10), x = 'MonthlyCharges',
                                                             y='TotalCharges', color = '#ec838a')

plt.title('Co-linearity of Monthly Charges and Total Charges \n',horizontalalignment="center", fontstyle = "normal", fontsize = "22", fontfamily = "sans-serif")
```

Co-linearity of Monthly Charges and Total Charges



```
#dropping TotalCharges:

dataset2 = dataset2.drop(columns = "TotalCharges")

#Revalidate Colinearity:

dataset2 = dataset[['gender', 'SeniorCitizen', 'Partner', 'Dependents',
                    'tenure', 'PhoneService', 'PaperlessBilling', 'MonthlyCharges']]

calc_vif(dataset2)
```

	variables	VIF
0	gender	1.879536
1	SeniorCitizen	1.323089
2	Partner	2.814574
3	Dependents	1.908533
4	tenure	3.287603
5	PhoneService	5.963240
6	PaperlessBilling	2.745897
7	MonthlyCharges	7.453993

```
[ ] #Applying changes in the main dataset:
```

```
dataset = dataset.drop(columns = "TotalCharges")
```

```
[ ] #Encode Categorical data
```

```
#Incase if user_id is an object:
```

```
identity = dataset["customerID"]
```

```
dataset = dataset.drop(columns="customerID")
```

```
# convert rest of categorical variable into dummy
```

```
dataset= pd.get_dummies(dataset)
```

```
#Rejoin userid to dataset (column concatenation)
```

```
dataset = pd.concat([dataset, identity], axis = 1)
```

```
#Step 11: Split dataset into dependent and independent variables--
```

```
#identify response variable:
```

```
response = dataset["Churn"]
```

```
dataset = dataset.drop(columns="Churn")
```

```
]
```

```
#Removing Identifiers
```

```
train_identity = X_train['customerID']
```

```
X_train = X_train.drop(columns = ['customerID'])
```

```
test_identity = X_test['customerID']
```

```
X_test = X_test.drop(columns = ['customerID'])
```

```
]
```

```
#Evaluating Model Results:
```

```
acc_results = []
```

```
auc_results = []
```

```
names = []
```

```
# set table to table to populate with performance results
```

```
col = ['Algorithm', 'ROC AUC Mean', 'ROC AUC STD', 'Accuracy Mean', 'Accuracy STD']
```

```
model_results = pd.DataFrame(columns=col)
```

```
i = 0
```

```
# evaluate each model using k-fold cross-validation
```

```
for name, model in models:
```

```
    kfold = model_selection.KFold(n_splits=10, random_state=0) # 10-fold cross-validation
```

```
    cv_acc_results = model_selection.cross_val_score( # accuracy scoring  
        model, X_train, y_train, cv=kfold, scoring='accuracy')
```

```
    cv_auc_results = model_selection.cross_val_score( # roc_auc scoring  
        model, X_train, y_train, cv=kfold, scoring='roc_auc')
```

```

acc_results.append(cv_acc_results)
auc_results.append(cv_auc_results)
names.append(name)
model_results.loc[i] = [name,
                        round(cv_auc_results.mean()*100, 2),
                        round(cv_auc_results.std()*100, 2),
                        round(cv_acc_results.mean()*100, 2),
                        round(cv_acc_results.std()*100, 2)
                        ]

i += 1

model_results.sort_values(by=['ROC AUC Mean'], ascending=False)

```

	Algorithm	ROC AUC Mean	ROC AUC STD	Accuracy Mean	Accuracy STD
0	Logistic Regression	84.12	1.65	74.60	1.26
1	SVC	83.64	1.68	79.98	1.08
4	Gaussian NB	81.82	1.79	68.99	1.46
6	Random Forest	81.72	2.02	78.47	1.57
2	Kernel SVM	79.66	2.12	79.85	1.08
3	KNN	77.04	2.38	75.77	1.09
5	Decision Tree Classifier	65.44	1.67	72.75	1.45

```
#Train & evaluate Chosen Model
```

```
#Fit Logistic Regression on the Training dataset:
```

```

classifier = LogisticRegression(random_state = 0, penalty = 'l2')
classifier.fit(X_train, y_train)

```

```
# Predict the Test set results
```

```
y_pred = classifier.predict(X_test)
```

```
#Evaluate Model Results on Test Set:
```

```

acc = accuracy_score(y_test, y_pred )
prec = precision_score(y_test, y_pred )
rec = recall_score(y_test, y_pred )
f1 = f1_score(y_test, y_pred )
f2 = fbeta_score(y_test, y_pred, beta=2.0)

```

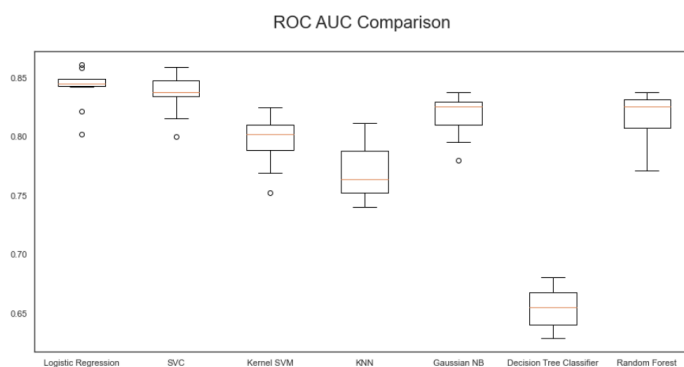
```

results = pd.DataFrame(['Logistic Regression', acc, prec, rec, f1, f2]),
                      columns = ['Model', 'Accuracy', 'Precision', 'Recall', 'F1 Score', 'F2 Score']

```

```
print (results)
```


	variables	VIF
0	gender	1.921286
1	SeniorCitizen	1.327766
2	Partner	2.815272
3	Dependents	1.921208
4	tenure	10.549667
5	PhoneService	7.976386
6	PaperlessBilling	2.814160
7	MonthlyCharges	13.988649
8	TotalCharges	12.570269



```
#Get the right parameters for the baseline models

#Identify optimal number of K neighbors for KNN Model:

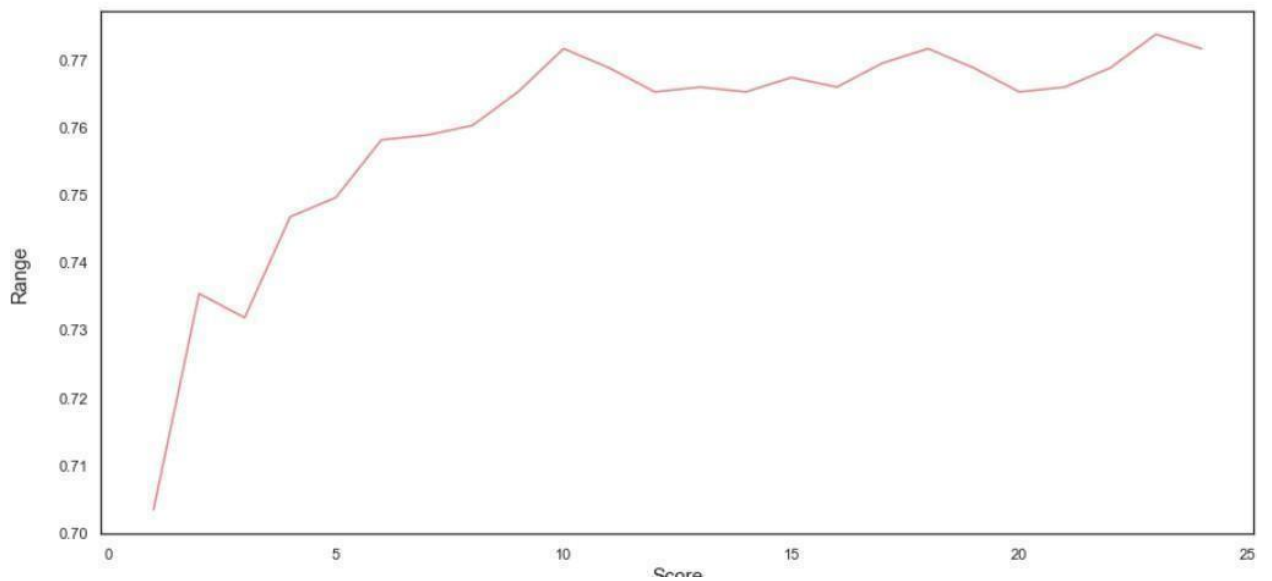
score_array = []
for each in range(1,25):
    knn_loop = KNeighborsClassifier(n_neighbors = each) #set K neighbor as 3
    knn_loop.fit(X_train,y_train)
    score_array.append(knn_loop.score(X_test,y_test))

fig = plt.figure(figsize=(15, 7))
plt.plot(range(1,25),score_array, color = '#ec838a')

plt.ylabel('Range\n',horizontalalignment="center",fontstyle = "normal", fontsize = "large", fontfamily = "sans-serif")
plt.xlabel('Score\n',horizontalalignment="center",fontstyle = "normal", fontsize = "large", fontfamily = "sans-serif")
plt.title('Optimal Number of K Neighbors \n',horizontalalignment="center", fontstyle = "normal", fontsize = "22", fontfamily = "sans-se
#plt.legend(loc='top right', fontsize = "medium")
plt.xticks(rotation=0, horizontalalignment="center")
plt.yticks(rotation=0, horizontalalignment="right")

plt.show()
```

Optimal Number of K Neighbors



```
[ ] #Identify optimal number of trees for Random Forest Model:

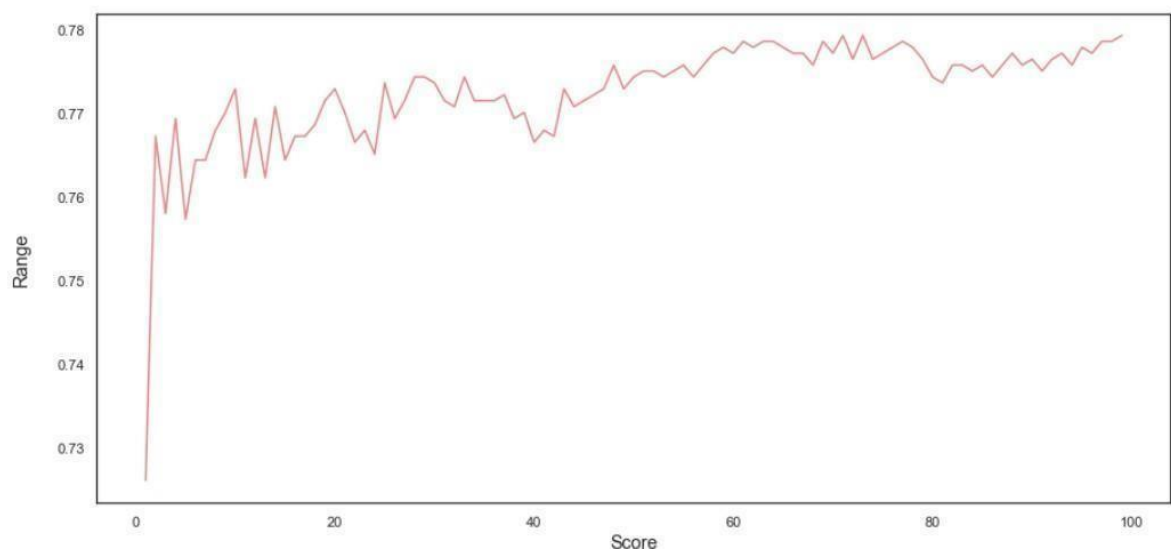
score_array = []
for each in range(1,100):
    rf_loop = RandomForestClassifier(n_estimators = each, random_state = 1)
    rf_loop.fit(X_train,y_train)
    score_array.append(rf_loop.score(X_test,y_test))

fig = plt.figure(figsize=(15, 7))
plt.plot(range(1,100),score_array, color = '#ec838a')

plt.ylabel('Range\n',horizontalalignment="center",fontstyle = "normal", fontsize = "large", fontfamily = "sans-serif")
plt.xlabel('Score\n',horizontalalignment="center",fontstyle = "normal", fontsize = "large", fontfamily = "sans-serif")
plt.title('Optimal Number of Trees for Random Forest Model \n',horizontalalignment="center", fontstyle = "normal", fontsize = "22", fontfamily = "sans-serif")
plt.legend(loc='top right', fontsize = "medium")
plt.xticks(rotation=0, horizontalalignment="center")
plt.yticks(rotation=0, horizontalalignment="right")

plt.show()
```

Optimal Number of Trees for Random Forest Model



```

#Train & evaluate Chosen Model

#Fit Logistic Regression on the Training dataset:

classifier = LogisticRegression(random_state = 0, penalty = 'l2')
classifier.fit(X_train, y_train)

# Predict the Test set results

y_pred = classifier.predict(X_test)

#Evaluate Model Results on Test Set:

acc = accuracy_score(y_test, y_pred )
prec = precision_score(y_test, y_pred )
rec = recall_score(y_test, y_pred )
f1 = f1_score(y_test, y_pred )
f2 = fbeta_score(y_test, y_pred, beta=2.0)

results = pd.DataFrame(['Logistic Regression', acc, prec, rec, f1, f2]),
                      columns = ['Model', 'Accuracy', 'Precision', 'Recall', 'F1 Score', 'F2 Score']

print (results)

```

```

# Fitting Logistic Regression to the Training set
classifier = LogisticRegression(random_state = 0)
classifier.fit(X_train, y_train)

# Predicting the Test set results
y_pred = classifier.predict(X_test)

#Evaluate results

acc = accuracy_score(y_test, y_pred )
prec = precision_score(y_test, y_pred )
rec = recall_score(y_test, y_pred )
f1 = f1_score(y_test, y_pred )
f2 = fbeta_score(y_test, y_pred, beta=2.0)

results = pd.DataFrame(['Logistic Regression', acc, prec, rec, f1, f2]),
                      columns = ['Model', 'Accuracy', 'Precision', 'Recall', 'F1 Score', 'F2 Score']

```

```

# Fitting SVM (SVC class) to the Training set:

classifier = SVC(kernel = 'linear', random_state = 0)
classifier.fit(X_train, y_train)

# Predicting the Test set results
y_pred = classifier.predict(X_test)

#Evaluate results

acc = accuracy_score(y_test, y_pred )
prec = precision_score(y_test, y_pred )
rec = recall_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred )
f2 = fbeta_score(y_test, y_pred, beta=2.0)

model_results = pd.DataFrame([['SVM (Linear)', acc, prec, rec, f1, f2]],
                             columns = ['Model', 'Accuracy', 'Precision', 'Recall', 'F1 Score', 'F2 Score'])

results = results.append(model_results, ignore_index = True)

```

#Step 15.4.3. K-Nearest Neighbours-----

```

# Fitting KNN to the Training set:

classifier = KNeighborsClassifier(n_neighbors = 22, metric = 'minkowski', p = 2)
classifier.fit(X_train, y_train)

# Predicting the Test set results
y_pred = classifier.predict(X_test)

#Evaluate results

acc = accuracy_score(y_test, y_pred )
prec = precision_score(y_test, y_pred )
rec = recall_score(y_test, y_pred )
f1 = f1_score(y_test, y_pred )
f2 = fbeta_score(y_test, y_pred, beta=2.0)

model_results = pd.DataFrame([['K-Nearest Neighbours', acc, prec, rec, f1, f2]],
                             columns = ['Model', 'Accuracy', 'Precision', 'Recall', 'F1 Score', 'F2 Score'])

results = results.append(model_results, ignore_index = True)

```

```

#Step 15.4.4. Kernel SVM-----

# Fitting Kernel SVM to the Training set:

classifier = SVC(kernel = 'rbf', random_state = 0)
classifier.fit(X_train, y_train)

# Predicting the Test set results
y_pred = classifier.predict(X_test)

#Evaluate results

acc = accuracy_score(y_test, y_pred )
prec = precision_score(y_test, y_pred )
rec = recall_score(y_test, y_pred )
f1 = f1_score(y_test, y_pred )
f2 = fbeta_score(y_test, y_pred, beta=2.0)

model_results = pd.DataFrame([[ 'Kernel SVM', acc, prec, rec, f1, f2]],
                             columns = [ 'Model', 'Accuracy', 'Precision', 'Recall', 'F1 Score', 'F2 Score'])

results = results.append(model_results, ignore_index = True)

```

```

#Decision Tree

# Fitting Decision Tree to the Training set:

classifier = DecisionTreeClassifier(criterion = 'entropy', random_state = 0)
classifier.fit(X_train, y_train)

# Predicting the Test set results
y_pred = classifier.predict(X_test)

#Evaluate results

acc = accuracy_score(y_test, y_pred )
prec = precision_score(y_test, y_pred )
rec = recall_score(y_test, y_pred )
f1 = f1_score(y_test, y_pred )
f2 = fbeta_score(y_test, y_pred, beta=2.0)

model_results = pd.DataFrame([[ 'Decision Tree', acc, prec, rec, f1, f2]],
                             columns = [ 'Model', 'Accuracy', 'Precision', 'Recall', 'F1 Score', 'F2 Score'])

results = results.append(model_results, ignore_index = True)

```



```

#Random Forest

# Fitting Random Forest to the Training set:

classifier = RandomForestClassifier(n_estimators = 72, criterion = 'entropy', random_state = 0)
classifier.fit(X_train, y_train)

# Predicting the Test set results
y_pred = classifier.predict(X_test)

#Evaluate results

from sklearn.metrics import confusion_matrix, accuracy_score, f1_score, precision_score, recall_score
acc = accuracy_score(y_test, y_pred )
prec = precision_score(y_test, y_pred )
rec = recall_score(y_test, y_pred )
f1 = f1_score(y_test, y_pred )
f2 = fbeta_score(y_test, y_pred, beta=2.0)

model_results = pd.DataFrame([['Random Forest', acc, prec, rec, f1, f2]],
                             columns = ['Model', 'Accuracy', 'Precision', 'Recall', 'F1 Score', 'F2 Score'])

results = results.append(model_results, ignore_index = True)

```

```

[ ] results = results.sort_values(["Precision", "Recall", "F2 Score"], ascending = False)

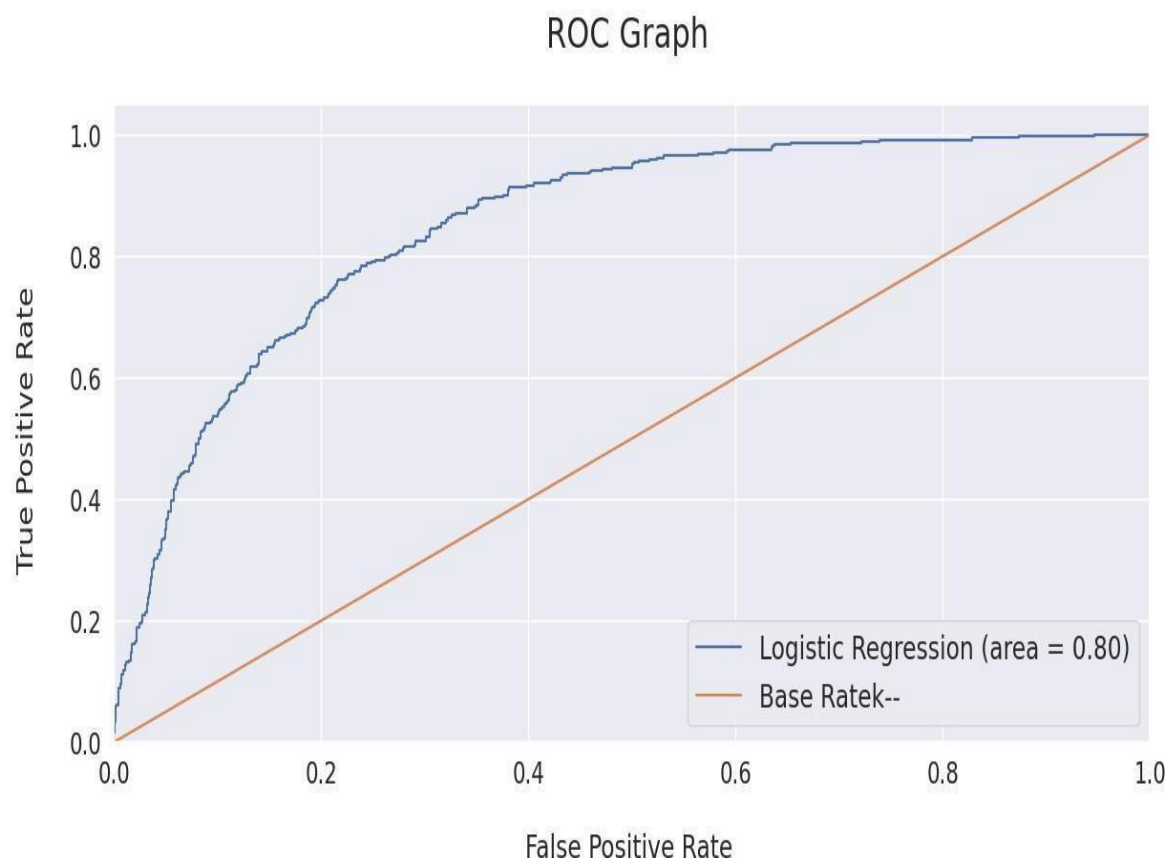
print (results)

```

	Model	Accuracy	Precision	Recall	F1 Score	F2 Score
0	Logistic Regression	0.803407	0.652038	0.556150	0.600289	0.573003
1	SVM (Linear)	0.803407	0.650155	0.561497	0.602582	0.577240
3	Kernel SVM	0.791341	0.637931	0.494652	0.557229	0.517917
6	Random Forest	0.779276	0.617100	0.443850	0.516330	0.470255
2	K-Nearest Neighbours	0.768630	0.570175	0.521390	0.544693	0.530468
5	Decision Tree	0.739532	0.508997	0.529412	0.519004	0.525199
4	Naive Byes	0.703336	0.467359	0.842246	0.601145	0.725806

```
[ ] dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7043 entries, 0 to 7042
Data columns (total 21 columns):
#   Column                Non-Null Count  Dtype
---  -
0   customerID            7043 non-null   object
1   gender                 7043 non-null   object
2   SeniorCitizen          7043 non-null   int64
3   Partner                7043 non-null   object
4   Dependents             7043 non-null   object
5   tenure                 7043 non-null   int64
6   PhoneService           7043 non-null   object
7   MultipleLines          7043 non-null   object
8   InternetService        7043 non-null   object
9   OnlineSecurity         7043 non-null   object
10  OnlineBackup           7043 non-null   object
11  DeviceProtection       7043 non-null   object
12  TechSupport            7043 non-null   object
13  StreamingTV            7043 non-null   object
14  StreamingMovies        7043 non-null   object
15  Contract               7043 non-null   object
16  PaperlessBilling       7043 non-null   object
17  PaymentMethod          7043 non-null   object
18  MonthlyCharges         7043 non-null   float64
19  TotalCharges           7032 non-null   float64
20  Churn                  7043 non-null   object
dtypes: float64(2), int64(2), object(17)
memory usage: 1.1+ MB
```



[67]

	features	coef	
12	InternetService_Fiber optic	0.326867	
32	Contract_Month-to-month	0.309004	
6	PaperlessBilling	0.165872	
28	StreamingTV_Yes	0.134275	
31	StreamingMovies_Yes	0.131698	
14	OnlineSecurity_No	0.118931	
37	PaymentMethod_Electronic check	0.113855	
23	TechSupport_No	0.095824	
10	MultipleLines_Yes	0.090564	
1	SeniorCitizen	0.079220	
17	OnlineBackup_No	0.053561	
22	DeviceProtection_Yes	0.049040	
20	DeviceProtection_No	0.023000	
19	OnlineBackup_Yes	0.017002	
5	PhoneService	0.001901	
9	MultipleLines_No phone service	-0.001901	
2	Partner	-0.008257	


```

lr_classifier = LogisticRegression(random_state = 0, penalty = 'l2')
lr_classifier.fit(X_train, y_train)
# Predict the Test set results
y_pred = lr_classifier.predict(X_test)
#probability score
y_pred_probs = lr_classifier.predict_proba(X_test)
y_pred_probs = y_pred_probs[:, 1]

```


```

[6] # Round 1:

# Select Regularization Method
import time
penalty = ['l1', 'l2']
# Create regularization hyperparameter space
C = [0.001, 0.01, 0.1, 1, 10, 100, 1000]
# Combine Parameters
parameters = dict(C=C, penalty=penalty)
lr_classifier = GridSearchCV(estimator = classifier,
                             param_grid = parameters,
                             scoring = "balanced_accuracy",
                             cv = 10,
                             n_jobs = -1)

t0 = time.time()
lr_classifier = lr_classifier.fit(X_train, y_train)
t1 = time.time()
print("Took %.2f seconds" % (t1 - t0))
lr_best_accuracy = lr_classifier.best_score_
lr_best_parameters = lr_classifier.best_params_
lr_best_accuracy, lr_best_parameters

```

 Took 6.69 seconds

```

# Round 2:
# Select Regularization Method
import time
penalty = ['l2']
# Create regularization hyperparameter space
C = [ 0.0001, 0.001, 0.01, 0.02, 0.05]
# Combine Parameters
parameters = dict(C=C, penalty=penalty)
lr_classifier = GridSearchCV(estimator = classifier,
                             param_grid = parameters,
                             scoring = "balanced_accuracy"
                             cv = 10,
                             n_jobs = -1)

t0 = time.time()
lr_classifier = lr_classifier .fit(X_train, y_train)
t1 = time.time()
print("Took %.2f seconds" % (t1 - t0))
lr_best_accuracy = lr_classifier.best_score_
lr_best_parameters = lr_classifier.best_params_
lr_best_accuracy, lr_best_parameters

```

```

> Took 1.76 seconds
(0.7187968786541289, {'C': 0.05, 'penalty': 'l2'})

```

```

lr_classifier = LogisticRegression(random_state = 0, penalty = 'l2')
lr_classifier.fit(X_train, y_train)
# Predict the Test set results
y_pred = lr_classifier.predict(X_test)
#probability score
y_pred_probs = lr_classifier.predict_proba(X_test)
y_pred_probs = y_pred_probs[:, 1]

```

```

72] final_results = pd.concat([test_identity, y_test], axis = 1).dropna()
final_results['predictions'] = y_pred
final_results["propensity_to_churn(%)"] = y_pred_probs
final_results["propensity_to_churn(%)"] = final_results["propensity_to_churn(%)"]*100
final_results["propensity_to_churn(%)"] = final_results["propensity_to_churn(%)"].round(2)
final_results = final_results[['customerID', 'Churn', 'predictions', 'propensity_to_churn(%)']]
final_results ['Ranking'] = pd.qcut(final_results['propensity_to_churn(%)'].rank(method = 'first'),10,labels=range(10,0,-1))
print (final_results)

```

	customerID	Churn	predictions	propensity_to_churn(%)	Ranking
5532	8174-LNWMW	0	0	1.94	9
5173	2480-SQIOB	0	0	33.65	4
7010	0723-DRCLG	1	1	79.61	1
6507	5708-EVONK	1	0	18.62	5
5985	3585-YNADK	0	0	4.15	8
...
3333	4573-JKNAE	0	0	3.14	9
3053	0960-HUWBM	0	0	6.46	7
6920	2595-KIWPV	0	0	1.78	9
4696	9128-CPXKI	0	0	9.64	7
2246	7181-BQYBV	1	1	74.84	1

[1409 rows x 5 columns]

Format - I

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY <small>(Deemed to be University u/s 3 of UGC Act, 1956)</small>		
Office of Controller of Examinations		
REPORT FOR PLAGIARISM CHECK ON THE DISSERTATION/PROJECT REPORTS FOR UG/PG PROGRAMMES (To be attached in the dissertation/ project report)		
1	Name of the Candidate (IN BLOCK LETTERS)	1. K.Rajarajan 2. Vishal Reddy
2	Address of the Candidate	No.2 elangoadigal street, potheri, kattankulathur, chengalpattu 603203
3	Registration Number	1. RA2011003010511 2. RA2011003010501
4	Date of Birth	1. 13/11/2002 2. 15/05/2002
5	Department	Computer Science and Engineering
6	Faculty	Engineering and Technology, School of Computing
7	Title of the Dissertation/Project	Anticipating Customer Churn in Telecommunication using ML Algorithm for customer retention
8	Whether the above project /dissertation is done by	<p>group : (Strike whichever is not applicable)</p> <p>a) If the project/ dissertation is done in group, then how many students together completed the project :</p> <p>b) Mention the Name & Register number of other candidates: Vishal Reddy.D (RA2011003010501)</p>
9	Name and address of the Supervisor / Guide	Dr.Priya S. Assistant professor Department of Computing Technologies SRM Institute of science and Technology Kattankulathur-603203 Mail ID: Priyas3@srmist.edu.in Mobile Number: +91 9965930862
10	Name and address of Co-Supervisor / Co- Guide (if any)	NIL
11	Software Used	Turnitin

12	Date of Verification	10/04/24		
13	Plagiarism Details: (to attach the final report from the software)			
Chapter	Title of the Chapter	Percentage of similarity index (including self citation)	Percentage of similarity index (Excluding self-citation)	% of plagiarism after excluding Quotes, Bibliography, etc.,
1	Introduction	1%	1%	0%
2	Literature Survey	1%	0%	0%
3	Architecture	0%	2%	1%
4	Result and Analysis	1%	2%	1%
5	Conclusion & Future scope	2%	1%	1%
Appendices		0%	1%	0%
I / We declare that the above information have been verified and found true to the best of my / our knowledge.				
Signature of the Candidate		Name & Signature of the Staff (Who uses the plagiarism check software)		
Name & Signature of the Supervisor/ Guide		Name & Signature of the Co-Supervisor/Co-Guide		
Name & Signature of the HOD				

PLAGARISM REPORT

Work 1

ORIGINALITY REPORT

2%

SIMILARITY INDEX

1%

INTERNET SOURCES

1%

PUBLICATIONS

2%

STUDENT PAPERS

PRIMARY SOURCES

1	"Recent Developments in Electronics and Communication Systems", IOS Press, 2023 Publication	1%
2	www.tnsroindia.org.in Internet Source	1%
3	www.ijraset.com Internet Source	1%
4	Submitted to University of California Riverside Student Paper	1%
5	www.scilit.net Internet Source	1%
6	S. Malini, C. Murugan, G. Dency Flora, Gowri Shangari E. "Embedded based Smart Accident Pre-Alert and Prevention System with Machine Learning", 2023 9th International Conference on Advanced Computing and Communication Systems (ICACCS), 2023 Publication	1%
7	Submitted to University of Leeds Student Paper	1%

PUBLICATION

ICDCECE - 2024 Acceptance Notification for Paper ID 802 External Inbox x



Microsoft CMT <email@msr-cmt.org>
to me, publicationchair ▾

Fri, Mar 22, 5:31PM ☆ ↶ ⋮

Dear Author(s),

Congratulations!

We are pleased to inform you that your paper has been Accepted for Oral presentation at the IEEE 3rd International Conference on Distributed Computing and Electrical Circuits And Electronics (ICDCECE-2024) in association with IEEE Bangalore Section during 26 & 27 April 2024 at Ballari Institute of Technology and Management, Ballari, Karnataka, India. The conference proceedings will be submitted to the IEEE Xplore® digital library.

NOTE: During Camera Ready Submission, please address the reviewer comments mentioned at the end of this email

Reviewer Comment:

Format the paper strictly according to the conference template available in the conference website.

Do not modify Layout margin & size of the template.

