Dynamic Pricing Models in Marketplace Environments; The Case of Ride Hailing Business

By

Evans Munyendo Ouma

051234

Submitted in Partial Fulfilment of the Requirements for the Degree of Master of Science in Data Science and Analytics at Strathmore University

Institute of Mathematical Sciences

Strathmore University

Nairobi, Kenya

June, 2025

Declaration and Approval

Declaration

I declare that this work has not been previously submitted and approved for the award of a degree by this or any other University. To the best of my knowledge and belief, the dissertation contains no material previously published or written by another person except where due reference is made in the dissertation itself.

Student's Name: Evans Munyendo Ouma

Sign:  Date: May 22nd, 2025

Approval

The dissertation of Evans Munyendo Ouma was reviewed and approved for examination by the following:

Dr. Anthony Kilili, PhD
Associate Professor, Institute of Mathematical Sciences,
Strathmore University

Dr. Godfrey Achono Madigu,
Dean, Institute of Mathematical Sciences,
Strathmore University

Prof. Bernard Shibwabo,
Director of Graduate Studies,
Strathmore University

Abstract

Classical economics theory proposes that with perfect information prices are efficient and will converge at the equilibrium of supply and demand curves. This is largely true for free markets. However, it is common for external agents e.g. governments via their central banks to propose and implement policies that exert an external force to influence prices. An example being changes to interest rates to control inflation, therefore control prices.

In on-demand and online marketplaces, prices would normally be influenced by forces of demand and supply. But in most cases the owners of the marketplaces (equivalent to governments in the previous case) would want to maximize revenues by taking advantage of market inefficiencies in real-time or near real-time. This has contributed to the rise of dynamic pricing agents that act on market information to make adjustments to prices.

Most dynamic pricing models' objective to maximize revenues have a short time horizon. This study seeks to expand that time horizon by incorporating customer retention on the platforms/marketplaces to also maximize future profits; by using machine learning models to predict price sensitivity of demand and supply agents based on past behaviour.

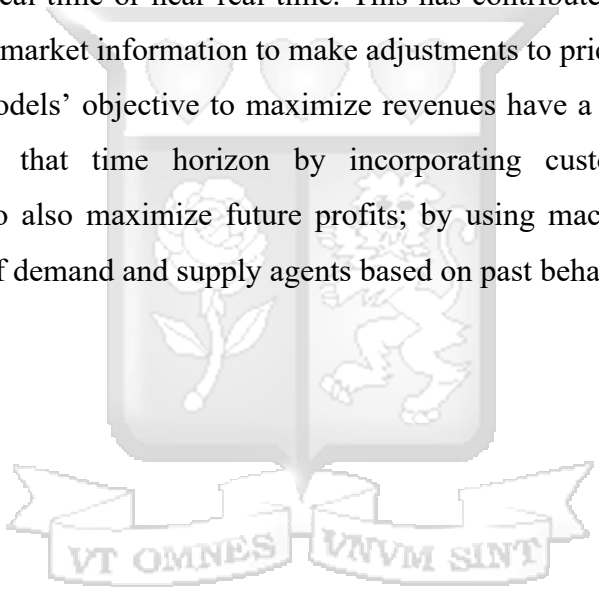

Table of Contents

List of Figures

List of Tables

List of Abbreviations

AUC	Area Under the Curve
LTV	Lifetime Value
MLP	Multi-Layered Perceptron
ROC	Receiver Operating Characteristics
SGD	Stochastic Gradient Decent
XGB	eXtreme Gradient Boosting

Definition of Terms

Algorithm	This is a step-by-step line written in human language as a guide on how a computer program will be written (Gurevich, 2015).
Area Under the Curve (AUC)	AUC is a performance indicator for binary classifiers models, and its value is the area under the Receiver Operating Characteristic (ROC) curve (Lavazza, Morasca, & Rotoloni, 2023)
Artificial Intelligence	Artificial intelligence (AI) is the capability of a computer or computer-controlled robot to perform tasks commonly associated with intelligent beings, such as reasoning, learning, and problem-solving (Copeland, 2024).
Data	Data are facts or information, especially when examined and used to find out things or to make decisions (Resnik, 2018).
Model	A model is an abstract representation that organizes elements of data and standardizes how they relate to one another and to the properties of real-world entities (Antoniou, 2021)
Receiver Operating Characteristics (ROC)	The ROC curve is the graphical representation obtained when conjoining these pairs in order of threshold in a plane, with false positive rates, and true positive rates on the x and y axes respectively (Lavazza, Morasca, & Rotoloni, 2023)

Acknowledgements

I would like to express my heartfelt gratitude to Dr. Anthony Kilili, my professor and supervisor, whose guidance and expertise have been invaluable throughout this journey. His course on machine learning not only deepened my understanding of the subject but also inspired me to pursue this research with rigor and enthusiasm.

I am also immensely grateful to Mr. Caesar Celsus, Head of BI and Intelligence at SafeBoda, and Mr. Simon Robertson, VP of Product and Analytics at SafeBoda, for their generosity in providing the data that served as the foundation of this study. Their collaboration and support have been instrumental in shaping the scope and depth of my research.

This thesis would not have been possible without their contributions, and I am deeply appreciative of the time, resources, and insights they shared.

Dedication

This thesis is dedicated to my colleagues and friends, whose discussions and insights have enriched my understanding and perspective.

I also dedicate this thesis to all aspiring researchers and practitioners in data science and machine learning, in the hope that it contributes meaningfully to the growing body of knowledge in the field.

Chapter 1: Introduction

1.1 Background to the Study

Over the last few years, the gig economy has seen a sharp rise in both supply and demand. This has been encouraged by a similar rise in and variety of platforms that create a common marketplace for both consumers and suppliers. In the e-commerce space the former giants eBay have been joined by Amazon, Alibaba, AliExpress, Jet.com just to name a few. These platforms would typically not offer products of their own but would allow merchants to offer services or products on their platforms

Marketplaces provide a common platform on which suppliers can reach a far much bigger pool of customers for their products and services. Customers on the other hand, gain access to a wide range of suppliers from which they can purchase product services from. The common commercial structure has been that the suppliers are allowed to list their prices, and the owners of the platform would collect commission from sales made. We do also have cases where customers are allowed to bid on products listed with the supplier having set a minimum price; e.g for the case of eBay. This has played into a scenario where suppliers have to optimize their prices to attract consumers while at the same time optimize their profits.

Although some of these platforms do not explicitly control the prices that suppliers want to offer to customers; most provide pricing analytics, serviced by automated agents, to help suppliers adjust their offering to try and beat the competition.

For the case of transport platforms like Uber, Lyft and their alternatives globally, we see a scenario where the price is controlled by the platform. In this case, there is a need to optimize both the driver and platform profits while at the same time ensuring retention of drivers (suppliers) and passengers (consumers). Unique for this marketplace is that the forces of demand and supply are quite dynamic, and the platforms are required to reach changes in real-time or near real-time.

For the case of SafeBoda, who will be the subject of this research, there exists a customer churn challenge. Prices, discounts and promos are adjustable but most happen after the fact, ie. after

analysis of customer and driver trends then effecting relatively static price changes. Customer retention initiatives e.g. promos and discounts, have been in the past applied with a blanket strategy where all customers get a price discount over a period of time. This typically has a positive effect where customers are engaged during the period of the market intervention, but customers would churn as soon as the promos are withdrawn.

Dynamic pricing in ride-hailing platforms can be framed as an intertemporal optimization problem, where prices are adjusted dynamically over time to maximize expected revenue while accounting for consumer demand fluctuations and resource constraints. By raising prices during peak-demand intervals and lowering them in off-peak periods, these algorithms effectively implement a form of intertemporal price discrimination that leverages temporal variations in customers' willingness to pay. This temporal differentiation can also be viewed through consumer surplus models: firms seek to appropriate a portion of the surplus consumers derive from time-flexible consumption opportunities by calibrating prices to shifting demand curves. Underpinning this approach is Muth's (1961) theory of rational expectations, which provides the foundation for anticipating future price movements and informing optimal price paths over multiple periods. In essence, dynamic pricing in marketplaces like ride-hailing balances the goal of maximizing intertemporal revenue with strategic consumer surplus extraction, all guided by predictive models of future demand and price expectations

Behavioural economics highlights that consumers' price perceptions are influenced by cognitive biases such as anchoring and fairness considerations. The anchoring effect suggests that individuals rely heavily on initial reference points when evaluating prices, which can significantly sway their willingness to pay. For instance, presenting a higher initial price can make subsequent lower prices appear more attractive, thereby influencing purchasing decisions. Moreover, perceptions of price fairness play a crucial role in consumer satisfaction and loyalty. Dynamic pricing strategies, if perceived as unjust or discriminatory, can lead to negative consumer reactions, including reduced trust and decreased purchase intentions (Haws & Bearden, 2006). Therefore, companies must carefully design pricing strategies that consider these psychological factors to maintain positive customer relationships.

An automated pricing model, which can be surfaced on the customer touch points as a discount or a promo, can be selectively channelled to customers that have been predicted to have a higher likelihood of churning. For this study, and from the context of SafeBoda's ride hailing business, a ride customer is deemed to have **churned** if they have not completed a ride 30 days after their most recent ride. Churn is used as an alternative to and opposite of **retention**. That is from a given point in time, and looking forward say 30 days, if the customer completed at least one ride in the period, they are deemed to have retained.

1.2 Problem Statement

Traditional pricing agents, especially for on-demand platforms like Uber and Lyft have typically relied on a limited set of variables to make adjustments to the price. Pricing strategies employed in the mobility platforms typically show an overreliance on demand data as measured by the number of orders at a given point of time in a given location. Although this is geared towards maximizing profits for the platform and the driver, customer retention is typically overlooked. The downside being that revenues will be maximized in the short windows in which this pricing strategy is applied, but long-term profitability will take a hit as customer retention suffers

This study seeks to investigate the performance of pricing models that maximize profits for the supply market agents and the platform, maximize retention for all market agents (suppliers and customers); in comparison to single dimensional pricing agents

1.3 Scope of the Study

This study will look at SafeBoda, an on-demand mobility platform running operations in Uganda. The data gathered looks at orders, customer profiles and marketplace characteristics over a period of 6 months starting from January 2022 to the end of June 2022.

The study seeks to introduce new models to the pricing strategies employed by marketplaces; this is especially true for platforms that dictate the pricing of the services/products offered. The models will make use of existing data on customers covering their transactions and engagements on the platform. The output of the model will be geared towards optimizing for both customer retention and revenue.

For this study, driver (supply agents) activity and transactions will not be considered.

1.4 Justification of the Study

Often trip pricing, normally price per kilometer, is normally determined beforehand and configured in the system. The setup is flexible to a good extent as it allows different pricing per geographies and price premiums or discounts depending on the day of week and time of day.
By design, this pricing strategy does not account for future revenue maximization but only current revenue objectives. The study does not seek to replace these pricing agents and the strategies they employ but seeks to augment them by adding a layer of price adjustments based on the likelihood of customers churning.

1.5 Research Objectives

The general objective of the study was:

i. To investigate the integration of customer retention metrics into dynamic pricing strategies in ride-hailing marketplaces.

The specific objectives for the study were:

i. To determine the key factors affecting the churn of customers (demand agents) in a ride hailing marketplace environment
ii. To incorporate customer retention metric as part of the inputs of dynamic pricing in a ride hailing marketplace
iii. To develop and quantify the performance of dynamic pricing models against traditional and often static pricing models in a ride hailing marketplace

1.6 Research Questions

i. What factors affect customer churn for a ride hailing marketplace?
ii. Can machine learning models accurately predict customer churn
iii. To what extent do dynamic pricing models outperform human agents in maximizing revenues and minimizing customer churn for a ride hailing business?

1.7 Significance of the Study

Dynamic pricing models have been extensively studied and implemented in the airline industry but have not made headway into the on-demand gig economy. In addition to contributing to the body of knowledge in this field, the results of this study have an impact on traditional e-commerce marketplaces too. With certain thresholds, dynamic pricing models can be deployed and employed by suppliers, allowing them to promptly react to changes in demand and supply

Chapter 2: Literature Review

2.1 Introduction

Dynamic pricing models have largely been popularized by low-cost airlines. Their distinctive feature, at least the one known to consumers, is the fluidity of the price of the tickets across a period of time. It is common for customers to notice that the price of the ticket was lower or higher within a few hours of them checking. Although the general trend is that prices will generally increase as the flight date approaches, it is not uncommon for prices to drop even a few hours to departure if the airplane is under capacity.

Another application of dynamic pricing has been employed in online bidding platforms. These models have been used to set the lower threshold or the starting bit of the item under sale; in marketplaces, this is after trying to estimate the strength of supply and demand. However, a shortfall of this strategy is that the starting bid price is set once, at the beginning and the platform has no chance to adjust it after the bidding starts. This leads to a situation where the item is oversubscribed or undersubscribed, with room for adjustments to maximize the selling price.

The on-demand economy has brought new opportunities for dynamic pricing models. An example of this being taxi platforms like Uber, Lyft and their equivalents around the world. The platform provides a marketplace for drivers (supply agents) and customers (demand) agents to interact. In most cases the platform controls the prices and would want to ensure retention of both the demand and supply agents, while at the same time maximize its profits and the revenues generated by its suppliers. As forces of demand and supply shift across the hours of day, days of the week and months, it is necessary for the platform to dynamically adjust the price to maximize revenue.

2.2 Theoretical Foundation of the Study

(Muth, 1961) puts forward a hypothesis that asserts that the economy generally does not waste information, and that expectations of prices depend specifically on the structure of the entire system. Short-period price variations in an isolated market with a fixed production lag of a commodity which cannot be stored can be simplified as a function of supply and demand; and that price converges the equilibrium of demand and supply curves.

The study seeks to propose dynamic pricing agents that identify market inefficiencies and leverage on the same to maximize profits.

Revenue management theory applies quantitative methods and optimization algorithms to perishable resources, such as ride-hailing seats, to maximize expected revenue over time (Talluri, Karaesmen, van Ryzin, & Vulcano, 2009). It integrates demand forecasting, inventory control and dynamic pricing across different customer segments to allocate limited capacity where it yields the highest marginal return (Talluri, Karaesmen, van Ryzin, & Vulcano, 2009).

Customer lifetime value (CLV) models conceptualize each customer as a stream of future net cash flows, discounted to their present value, thereby treating customer acquisition and retention expenditures as investments rather than expenses (Gupta, Lehmann, & Stuart, 2004). By forecasting churn probabilities and transactional behavior, these models optimize marketing spend and segment-specific strategies to maximize total customer equity (Gupta, Lehmann, & Stuart, 2004). This approach speaks to the long-term strategy in pricing when trying to optimize for customer churn, and future revenues

Game-theoretic pricing in two-sided marketplaces examines how platforms set fees on each side of the market—drivers and riders—while accounting for cross-side network externalities (Rochet & Tirole, 2003). Rochet and Tirole's framework shows that optimal price structures must balance participation incentives on both sides, since the utility of one side depends on the volume and quality of the other, leading to pricing formulas that internalize usage and membership externalities (Rochet & Tirole, 2003)

2.3 Empirical Literature Reviews

2.3.1 Definition

Dynamic pricing agents are software programs or algorithms that use machine learning and artificial intelligence techniques to adjust prices dynamically in real-time based on changes in supply and demand. These agents are typically used in e-commerce and other online

marketplaces to maximize revenue by setting prices that are most likely to attract buyers while also ensuring that the seller makes a profit.

From the context of an e-commerce platform where these pricing agents are widely employed, they take into account a wide range of variables, including the current inventory levels, historical sales data, competitor pricing, seasonality, and customer behavior patterns. These datasets will typically be historically generated by the eCommerce platform itself. They can also analyze data from external sources such as weather forecasts, social media trends, and economic indicators to adjust prices in response to changes in the market.

(Kephart, Hanson, & Greenwald, 2000) proposed a future in which the global economy and the Internet will merge, evolving into an information economy with billions of economically motivated software agents that exchange information, goods and services with humans and other agents.

Dynamic pricing has also been described as the study of determining optimal selling prices of products or services, in a setting where prices can easily and frequently be adjusted. This has seen application by vendors who sell on the internet, and by brick-and-mortar stores that make use of digital price tags. In both cases, digital technology has made it possible to continuously adjust prices to changing circumstances, with minimal cost implications (den Boer, 2015).

The goal of a majority of dynamic pricing agents has been to optimize pricing decisions to achieve a balance between profitability and market share, while also delivering a positive customer experience. By adjusting prices in real-time, these agents can help businesses stay competitive and responsive to changing market conditions, while also maximizing their revenue potential.

2.3.2 Usage

Pricing Models in e-Commerce

Dynamic pricing models have been implemented in traditional retail by making adjustments to price of inventory after analytics demand. By analyzing the deterministic version of different versions of the basic problem, upper bounds on the expected revenue were obtained and insights into the form of near-optimal policies. The strongest conclusion from the results is that using

simple fixed-price policies appears to work surprisingly well in many instances. This is encouraging since the optimal dynamic policies are quite jittery and require constant price adjustments, an undesirable characteristic in practical applications (Gallego & van Ryzin, 1994). In this case the model was allowed to make sales even in the cases where there was no inventory, and make price adjustments on the fly. However, the optimal system was achieved when thresholds were imposed on the pricing model, and the prices would be adjusted step-wise, that is the closest between two neighbouring thresholds.

In most cases, automated pricing models will not operate in a full information environment, that is, the pricing agent has to make assumptions and/or estimations of other variables that it does not have access to. This can be the case of competitor prices, demand and supply. The practical application of these models often faces challenges due to incomplete market information, such as competitor pricing or real-time demand and supply fluctuations. Computational limitations can further hinder the ability of pricing agents to react promptly to market dynamics. Still, even in the unlikely event that the agent has access to full knowledge of the of the environment is operates, integrating this information in the pricing model may lead to delays (e.g. due to computational capacity) where the agent is not able to react to the market forces in time to make an impact.

In a study of simulation-based approach to dynamic pricing, Goal-Directed and Derivative-Following strategies were found to be computationally straightforward, and robust under extremely different market conditions. Under every case presented, excluding the situation of 100% comparison-shopping, the strategies managed to adjust prices in the direction of learning the changing demand in the marketplace, without knowing the true buyer demand, competitors' prices, or even the number of other agents in the marketplace (DiMicco, Greenwald, & Maes, 2003)

As a pioneer of dynamic pricing strategies, RyanAir has remained competitive by taking advantage of “latent demand”. Latent demand is characterized by a different customer’s willingness to pay and a distinct elasticity to prices compared with the attitude of a typical passenger. The price discrimination techniques used by full cost carriers are based on a system of different fare classes, a complex system of discounts with limited access, the use of customer

loyalty schemes and of overbooking techniques (Malighetti, Paleari, & Redondi, 2007). Pricing strategies employed by Ryanair have seen it remain competitive over the years despite the low price of tickets. And especially in Europe, it is not uncommon for passengers to find tickets for as low as €20 for a trip that would otherwise cost €100. Although most of the data the pricing model relies on the organizations internal data (customer bookings, schedules, price thresholds and flight capacity), recent trends in airline pricing strategies have made use of competitor prices. This has been made possible by web scraping tools and aggregator websites that make tickets for the different airlines public and easy to source.

Many automated pricing models have been developed to focus on the demand side of the market as customer demand has been found to be more sensitive to price changes. Customer demand is then modelled to allow for short-term prediction. After demand behaviour of customers is learnt, how demand response to prices in the future can be forecasted and the optimal pricing policy for the next planning period will be given by an optimizer for dynamic pricing, such that the expected revenue is the greatest of revenues generated by all possible pricing policies. Therefore, an efficient demand learning model which gives accurate predictions is critical to pricing decisions, especially to dynamic pricing (Li, Yao, & Gao, 2010)

Churn Prediction in Machine Learning

Within the churn-prediction literature, systematic reviews highlight the dominance of tree-based ensembles and deep-learning architectures for classifying at-risk customers, emphasizing predictive accuracy improvements over interpretability (Soumi, 2022). Comparative studies reveal that while neural networks achieve top performance on large datasets, simpler models (e.g., logistic regression) often match accuracy on moderate-sized telecom and financial services data with far lower computational cost (Kriti, 2019) . However, these investigations tend to treat behavioural features as black-box inputs and seldom connect feature importance back to economic retention concepts such as marginal retention cost (Payam, Sogand , & Cosimo, 2025).

Hybrid Models Integrating Behaviour and Pricing

A growing body of research seeks to bridge pricing optimization with behavioural insights, using reinforcement learning or sentiment analysis to modulate prices based on fairness perceptions

and reference-price effects (Chenavaz & Dimitrov , 2025). Other hybrid approaches combine game-theoretic two-sided market models with demand-learning algorithms to internalize network-externality and retention-cost considerations, yet they rarely incorporate predictive churn signals at the individual level (Shin, 2023). Most recently, multi-modal frameworks leverage real-time engagement metrics (e.g., usage frequency, wallet top-ups) within the dynamic pricing loop, demonstrating improved customer lifetime value but still lacking a unified theory linking consumer surplus decay to price path decisions (Cheppala & Lakshya, 2024).

2.3.3 Advantages of Dynamic Pricing

Dynamic pricing agents enable firms to capture greater revenue by continuously adjusting prices to reflect prevailing market conditions, leveraging real-time or near real-time analyses of demand and supply dynamics to set optimal fare levels. By responding swiftly to shifts in consumer preferences and competitor actions, companies maintain a competitive edge, attracting new customers while preserving or expanding their market share. Moreover, dynamic pricing contributes to more efficient inventory management: for businesses offering tangible goods, price adjustments anchored to current stock levels serve as a mechanism to balance supply and demand, thereby mitigating the risks of both stockouts and excess inventory. Finally, the automation afforded by machine learning and artificial intelligence algorithms streamlines the pricing process itself—replacing manual rate-setting with instantaneous, data-driven decisions that save considerable time and effort while ensuring that price points remain closely aligned with evolving market realities.

2.3.4 Disadvantages of Dynamic Pricing

Dynamic pricing systems, however, entail substantial upfront and ongoing investments that can strain the budgets of smaller firms; the development, integration, and continuous maintenance of sophisticated algorithms and the requisite data infrastructure may prove cost-prohibitive. Furthermore, frequent or uneven price fluctuations can provoke customer dissatisfaction, as consumers who observe peers paying lower rates for identical services may perceive the system as unfair. This perception is exacerbated by the inherent opacity of many dynamic pricing models—their complex, algorithmic nature often leaves customers with little understanding of

why prices change, undermining trust and eroding confidence in the fairness of the pricing process.

2.4 Gaps in Research

In classical economic theory, a free isolated market will achieve a price at the equilibrium of forces of demand and supply. Dynamic pricing models, whose goals are typically to maximize profits, will look for and conjure up imbalances in the market to achieve this. However, most pricing agents will maximize profits or revenues in the near-term. This study seeks to improve dynamic pricing models to also improve future profits by incorporating customer (demand agents) retention.

Chapter 3: Research Methodology

3.1 Introduction

The purpose of this section is to lay out the overall methodology of the study, including the design and plan of data collection, analysis and interpretation. Covered here is the rationalization of the variables selected for the research, their descriptive and empirical associations linking them to measures of supply and demand; population and sample selection, techniques used for data analysis and evaluation of model performance and ethical considerations.

Predictive modelling in churn analysis focuses on identifying patterns in historical customer behaviour to forecast who is most likely to cancel their service in the future. These models leverage machine learning algorithms that optimize for accuracy in unseen data, selecting features and tuning parameters solely to improve predictive performance rather than to uncover underlying causal mechanisms. As a result, predictive approaches excel at ranking customers by churn propensity and enabling targeted interventions, but they do not, by themselves, explain why customers churn.

In contrast, causal inference methods aim to estimate the effect of specific actions or interventions such as a pricing change or promotional offer on churn by modelling counterfactual scenarios and controlling for confounding factors. While causal models answer "what if" questions about how changing one variable would alter outcomes, the objective in this thesis is to build a robust predictor of churn rather than to infer the causal impact of individual drivers on retention. Consequently, the methodology emphasizes maximizing out-of-sample prediction accuracy over isolating causal pathways.

3.2 Population and Sampling

For this research, data gathered is required to be sequential as pricing agents would require full information of the environment they operate in. As a result, the sample is determined by a time horizon as opposed to random sampling.

3.3 Data Collection Methods

Data supporting this research was collected from the database as sources by the platform from the driver and passenger applications. The data covers 8 months from 2022-02-01 to 2022-10-30 for all monetized customer services in the ecosystem, limited to Uganda only

The data set can be largely split into these groupings

i. Customer profile: 168,000 records

Limited customer profile information, covering date registered on the platform, boolean for if they have completed at least one ride and date of their first completed ride

ii. Customer engagement: 13,407,030 records

Counts and monetary value of service usage on the platform aggregated on date and customer level

iii. Ride data 10,167,103

A single record covers information about the ride including customer and driver matched, timestamp of request, final state of the ride (completed, or cancelled), price and payment type.

From these secondary data points are computed from aggregations and disaggregation of primary variables

3.4 Operationalization of the Variables

All the variables in this study are quantitative in nature

Outcome variables

Price per kilometre, which is a variable in on-demand economies

Profit, which is a function of price and costs that may be in form of discounts and promos

Independent variables

Platform event data including customer orders (demand)

Customer profiles including value and volume of previous orders

3.5 Exploratory Data Analysis

3.5.1 Variables Definition

Table 3.1: Customer profile variables

Variable	Description
user_id	Unique customer identifier
date_joined	Date of customer registration
ride_first_date	Date of the first paid completed ride by the customer

Daily aggregation of customer activity (counts)

Table 3.2: Count of customer platform activity by service

Variable	Description
date	Date of activity
user_id	Unique customer identifier
topup	Count of credit loads into the the SafeBoda wallet
bike	Count of bike rides completed
bike_cash	Count of bike rides completed and paid with cash
bike_cashless	Count of bike rides completed and paid via the Safeboda wallet
send	Count of parcel rides completed
car	Count of car rides completed
airtime	Count of airtime purchases
data	Count of data bundle purchased
bill	Count of bill payments
p2v	Count of vendor payments (passenger/customer to vendor)
o2p	Count of office/company to passenger/customer transactions e.g refunds
s2p	Count of safeboda/driver to passenger transfers
p2p	Count of passenger to passenger transfers (customer to customer)
p2a	Count of passenger /customer to agent transfers (cash withdrawal)
a2p	Count of agent to passenger/customer transfers (cash deposit)
p2b	Count of passenger/customer to bank transfers
p2mm	Count of passenger/customer to mobile money transfers

Daily aggregation of customer activity (value in UGX)

Table 3.3: Value of customer platform activity by service

Variable	Description
date	Date of activity
user_id	Unique customer identifier
topup	Value of credit loads into the the SafeBoda wallet
bike	Value of bike rides completed
bike_cash	Value of bike rides completed and paid with cash

bike_cashless	Value of bike rides completed and paid via the Safeboda wallet
send	Value of parcel rides completed
car	Value of car rides completed
airtime	Value of airtime purchases
data	Value of data bundle purchased
bill	Value of bill payments
p2v	Value of vendor payments (passenger/customer to vendor)
o2p	Value of office/company to passenger/customer transactions e.g refunds
s2p	Value of safeboda/driver to passenger transfers
p2p	Value of passenger-to-passenger transfers (customer to customer)
p2a	Value of passenger /customer to agent transfers (cash withdrawal)
a2p	Value of agent to passenger/customer transfers (cash deposit)
p2b	Value of passenger/customer to bank transfers
p2mm	Value of passenger/customer to mobile money transfers

Ride data: all rides requested and their final states (failed, cancelled or completed)

Table 3.4: Customer ride request data

Variable	Description
user_id	Unique customer identifier
driver_id	Unique driver identifier
trip_id	Unique trip/ride identifier
date	Date of the request
requested_at	Timestamp of the request
ended_trip_at	Timestamp of ride completion
current_state	Final state of the ride
distance_charge	Value of the ride as calculated from distance covered
duration_charge	Value of the ride as calculated from trip duration
total	Total value in UGX charged
payment_type	Cash or credit (wallet, business, or promo)
cash_paid	Value of the ride paid in cash
cashless_paid	Value of the ride paid from the digital wallet
business_paid	Value of the ride paid from the business wallet
promo_paid	Value of the ride paid from the promo wallet
trip_distance	Distance in meters covered during the trip
trip_duration_minutes	Time in minutes taken by the trip
trip_rating	Value ranging from 1-5 assigned to the driver by the customer

Periodic Customer Profile
An aggregation of customers activity over a given period of time

Table 3.5: Aggregation of customer ride activity data

Variable	Description
user_id	Unique customer identifier
active_weeks	Number of weeks in the measurement period where the customer has at least one completed ride
requests	Number of attempts by a customer to match to a driver
cancelled_rides	Number of rides that the customer cancelled
completed_rides	Number of rides that the customer completed
ride_value	Value in UGX of completed rides
ride_cash_value	Value in UGX of completed rides paid in cash
ride_cashless_value	Value in UGX of completed rides paid via the digital wallet
requests_per_week	Average number of attempts to match a driver in a week
completed_rides_per_week	Average number of completed rides in a week
ride_value_per_week	Average value of rides in UGX of completed rides in a week
cash_rides_per_week	Average number of rides paid in cash, completed in a week
cashless_rides_per_week	Average number of rides paid in via the digital wallet, completed in a week

3.5.2 Exploratory analysis of key variables

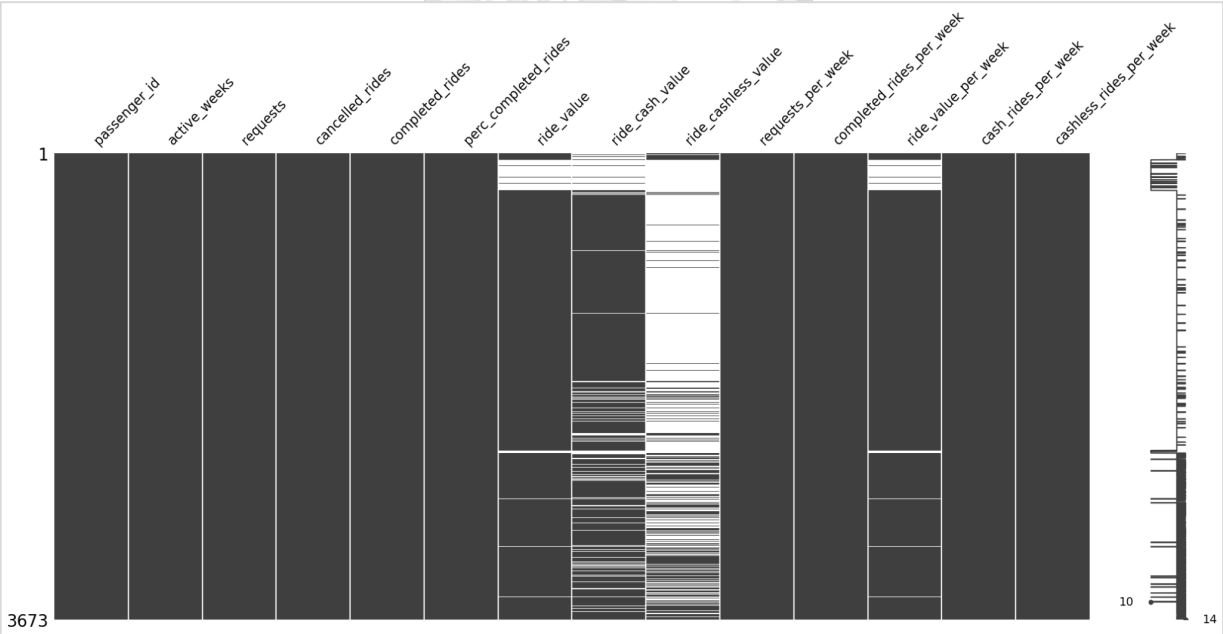

Figure 3.1: Map of data set, data completeness per column (select columns)

Table 3.6: Descriptive summary statistics for measures (select columns)

	count	mean	std	min	25%	50%	75%	max

active_weeks	3673	8.973863	8.012407	1	1	7	16	26
requests	3673	14.72992	29.86321	1	2	5	14	634
cancelled_rides	3673	2.76477	9.638492	0	0	0	2	319
completed_rides	3673	10.09611	18.91109	0	1	3	10	226
perc_completed_rides	3673	0.747889	0.32356	0	0.5	1	1	1
ride_value	3410	46292.23	80923.79	0	6500	18000	49500	1103000
ride_cash_value	3085	43004.05	72633.29	0	7000	17500	46000	1089000
ride_cashless_value	948	26332.49	64946.28	0	4000	8200	22000	805000
requests_per_week	3673	1.607106	1.708312	0.08	0.7	1	2	31
completed_rides_per_week	3673	1.091848	1.083813	0	0.4	1	1.25	12
ride_value_per_week	3410	5264.541	5969.977	0	2000	3605.556	6750	162500
cash_rides_per_week	3673	0.869946	0.93705	0	0.222222	0.666667	1	8.88
cashless_rides_per_week	3673	0.219513	0.690335	0	0	0	0.05	12

From the inspection of the summary statistics, it is evident that the data has some outliers in the data set. This would likely skew the results of analysis and inference that is to be drawn from the data. As such, outliers will be removed in the next steps before further analysis and data modelling

Table 3.7: Correlation map of measures (select columns)

	active_weeks	requests	cancelled_rides	completed_rides	perc_completed_rides	ride_value	ride_cash_value	ride_cashless_value	requests_per_week	completed_rides_per_week	ride_value_per_week	cash_rides_per_week	cashless_rides_per_week
active_weeks	1.0000	0.5329	0.3417	0.5586	-0.0360	0.5416	0.5213	0.2974	0.0225	0.0343	-0.0765	0.0905	-0.0681
requests	0.5329	1.0000	0.8237	0.9274	-0.0953	0.9009	0.8024	0.5881	0.6558	0.5879	0.4068	0.5205	0.2115
cancelled_rides	0.3417	0.8237	1.0000	0.5792	-0.2353	0.5847	0.5196	0.3734	0.5689	0.3345	0.2505	0.2901	0.1324
completed_rides	0.5586	0.9274	0.5792	1.0000	0.0527	0.9541	0.8569	0.6370	0.5845	0.6637	0.4461	0.5948	0.2262
perc_completed_rides	-0.0360	-0.0953	-0.2353	0.0527	1.0000	-0.0596	-0.0431	-0.0798	-0.1992	0.2535	0.0880	0.2447	0.0632
ride_value	0.5416	0.9009	0.5847	0.9541	-0.0596	1.0000	0.9032	0.6739	0.5865	0.6237	0.5261	0.5492	0.2121
ride_cash_value	0.5213	0.8024	0.5196	0.8569	-0.0431	0.9032	1.0000	0.0561	0.5570	0.5745	0.4734	0.6444	0.0098
ride_cashless_value	0.2974	0.5881	0.3734	0.6370	-0.0798	0.6739	0.0561	1.0000	0.4415	0.4907	0.5320	0.0367	0.5644
requests_per_week	0.0225	0.6558	0.5689	0.5845	-0.1992	0.5865	0.5570	0.4415	1.0000	0.8081	0.6326	0.5763	0.4765
completed_rides_per_week	0.0343	0.5879	0.3345	0.6637	0.2535	0.6237	0.5745	0.4907	0.8081	1.0000	0.7336	0.7683	0.5100
ride_value_per_week	-0.0765	0.4068	0.2505	0.4461	0.0880	0.5261	0.4734	0.5320	0.6326	0.7336	1.0000	0.5525	0.3696
cash_rides_per_week	0.0905	0.5205	0.2901	0.5948	0.2447	0.5492	0.6444	0.0367	0.5763	0.7683	0.5525	1.0000	-0.1508
cashless_rides_per_week	-0.0681	0.2115	0.1324	0.2262	0.0632	0.2121	0.0098	0.5644	0.4765	0.5100	0.3696	-0.1508	1.0000

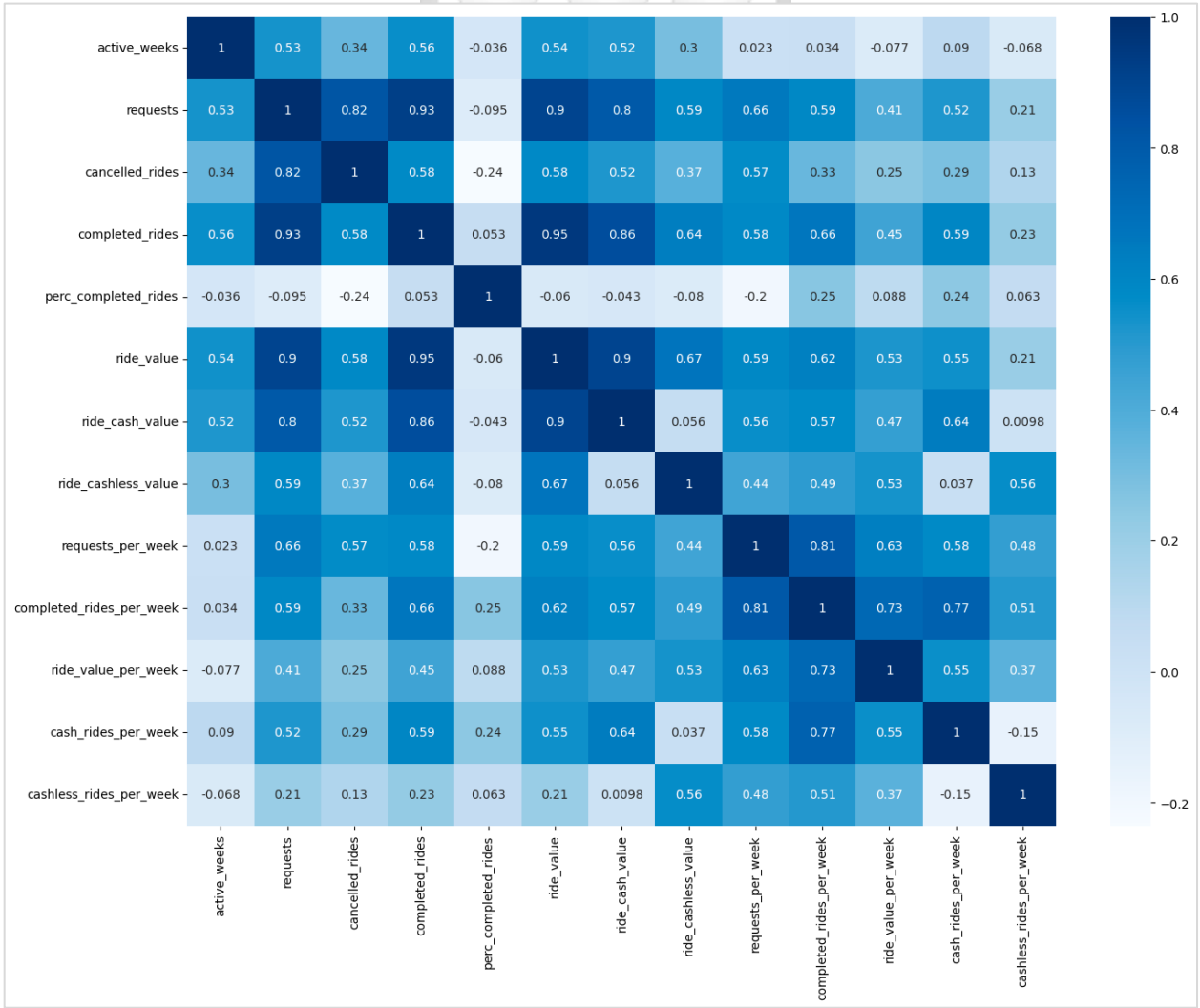

Figure 3.2: Correlation heatmap of features (selected features)

From the correlation data and heatmaps, there is a strong correlation between variables within the customer profile group and separately within the ridership group of variables. Outside of the two groups of data, there's minimal correlation. This is an early indicator that customer profiles, in their raw forms, may be a good predictor of ridership behaviour of passengers.

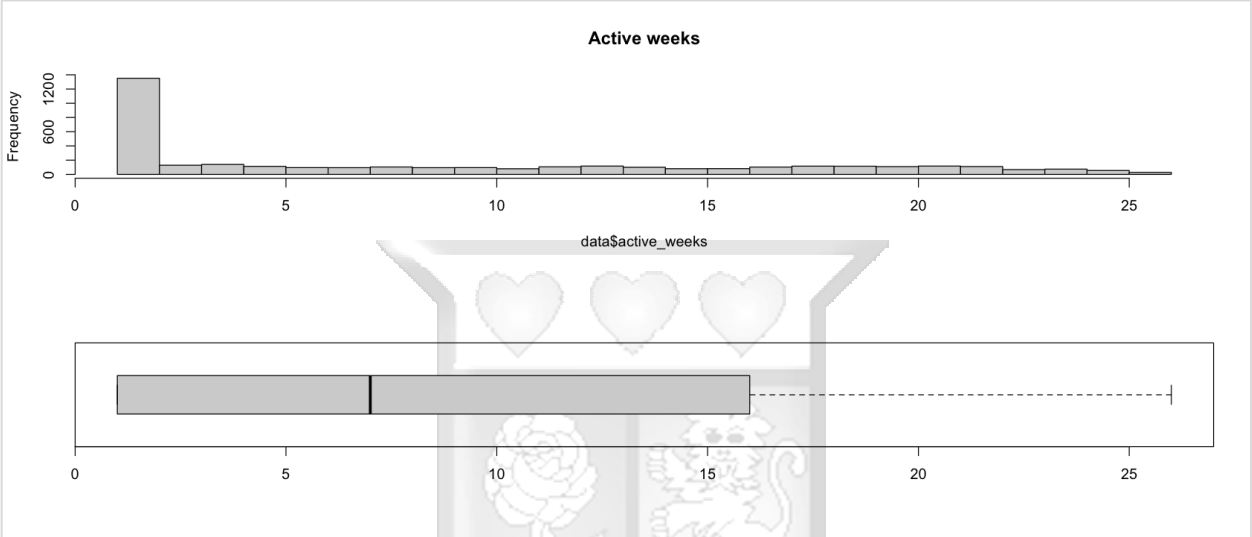

Figure 3.3: Histogram and boxplot of active weeks

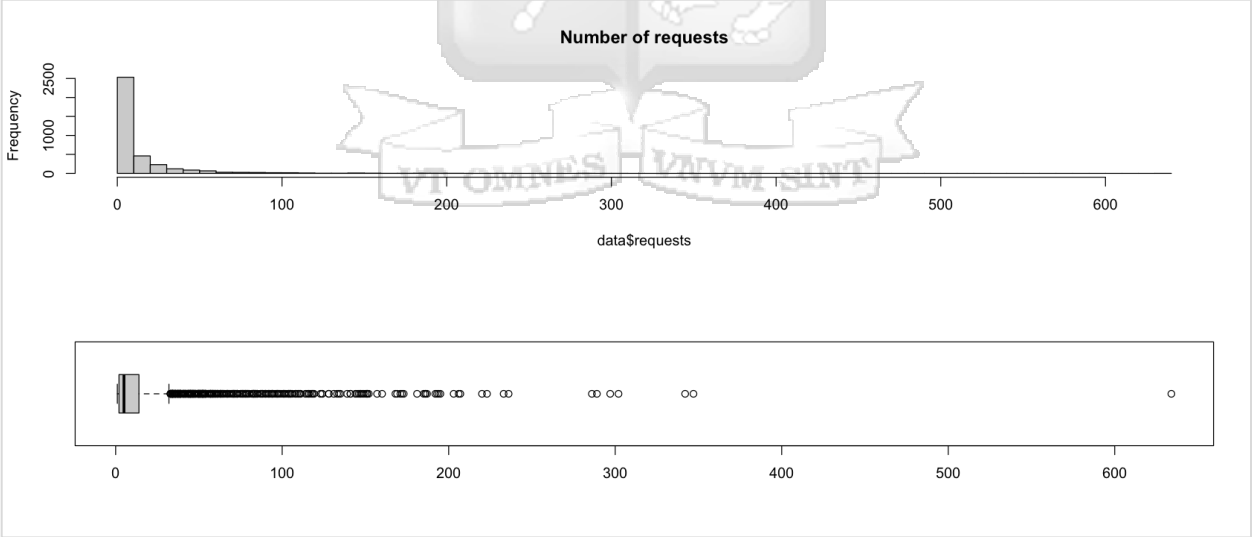

Figure 3.4: Histogram and boxplot of number of requests

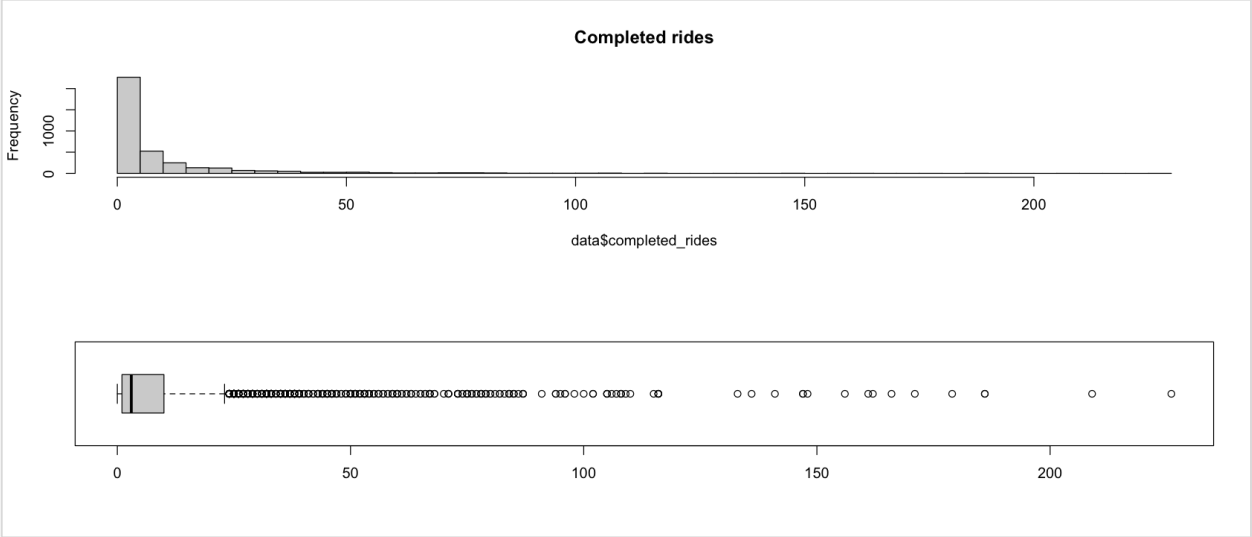

Figure 3.5: Histogram and boxplot of completed rides

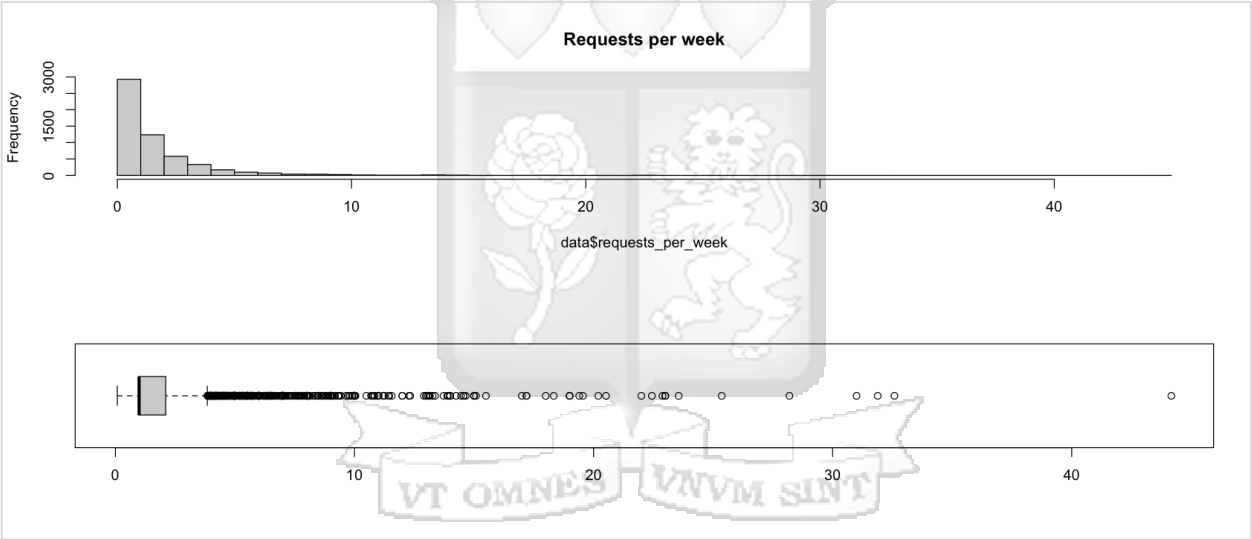

Figure 3.6: Histogram and boxplot of requests per week

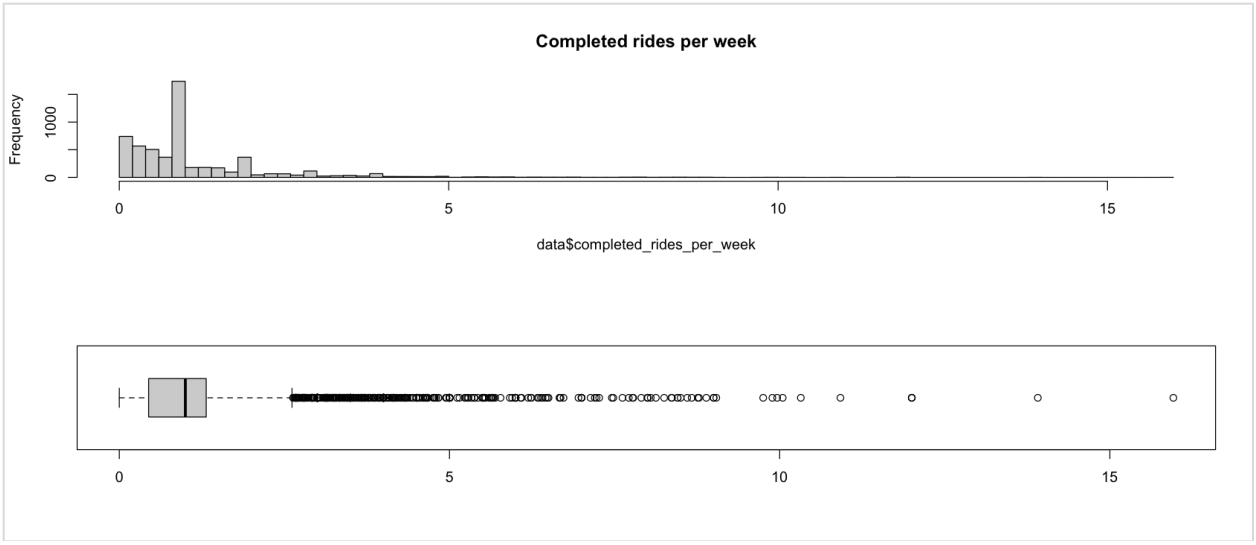

Figure 3.7: Histogram and boxplot of completed rides per week

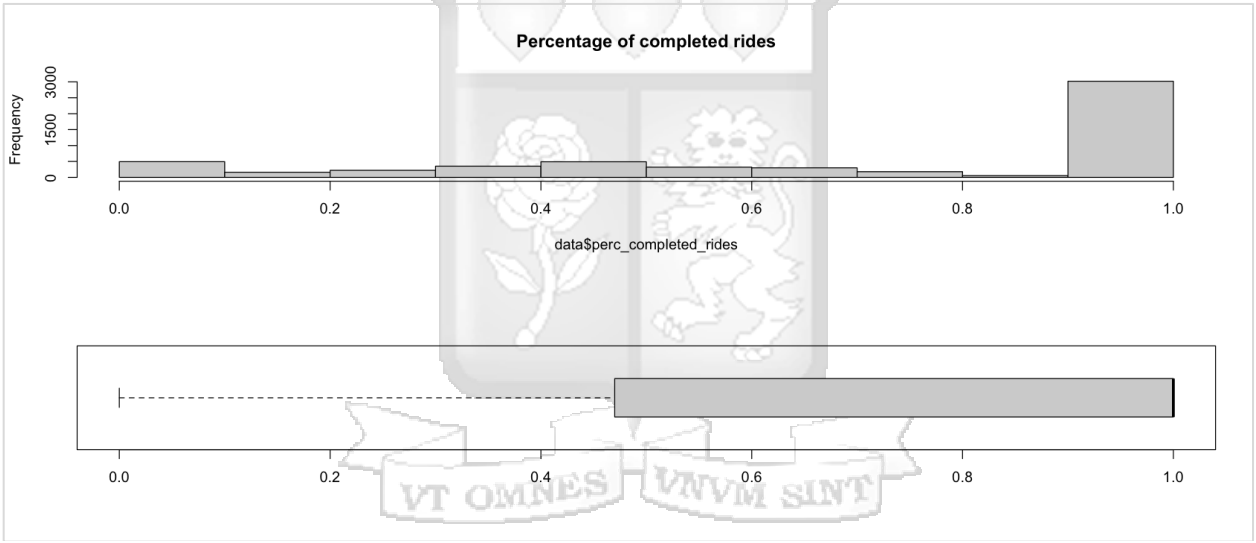

Figure 3.8: Histogram and boxplot of completed rides per week

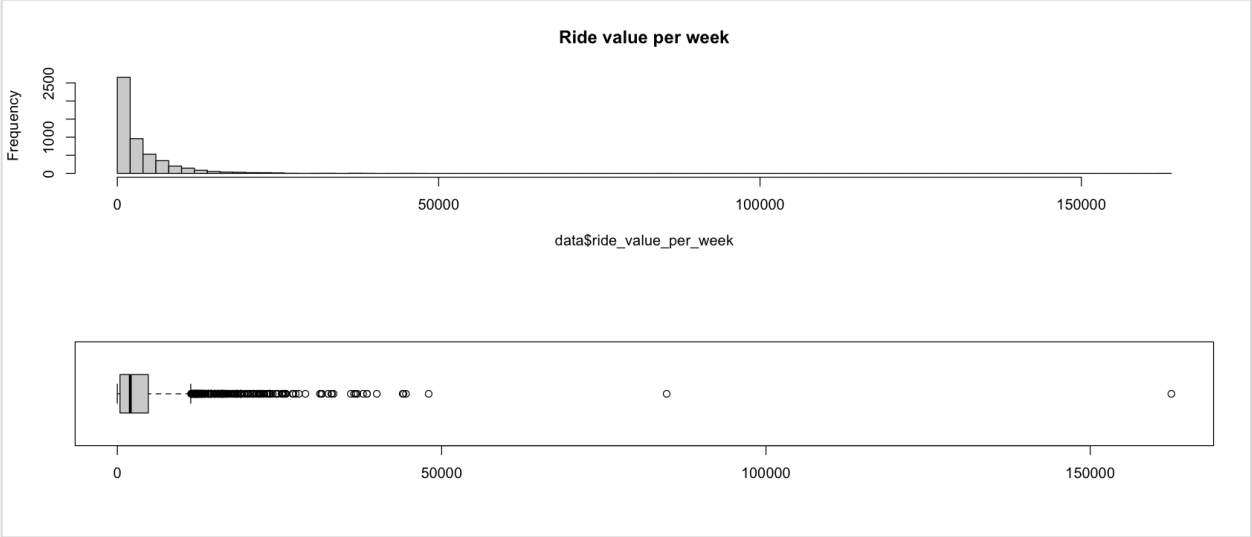

Figure 3.9: Histogram of ride value per week

An analysis of the key measures on histograms shows a consistent shape in the data, where variables exhibit long tails. This is key in advising the choice of statistical models to be applied on the data

3.6 Ethical Considerations

New rules of the data economy are straightforward, all of them derived from the basic principle that personal data is an asset held by the people who generate it. But each rule entails the breaking of entrenched habits, routines and networks (Rahnama & Pentland, 2022). In the current data economy, data is exchanged between systems that house algorithms to churn insight out the data. Regulations in most jurisdictions require that identifiable data is not shared outside of the organization without explicit consent by the owner of the data, in most cases these are the customer.

The co-design of algorithms and data can facilitate the process of insight extraction by structuring each to better meet the needs of the other. As a result, rather than moving data around, the algorithms exchange non-identifying statistics instead (Rahnama & Pentland, 2022). Data for this research is anonymized and cannot be used to identify customers.

3.7 Steps in Research Methodology

The analysis of the data will attempt to answer the following questions

i. Can machine learning be employed to accurately **predict the customer churn** based on historical purchasing patterns?

ii. Does employing the use of machine learning agents in determining whether or not do apply a discount to a price outperform human agents in maximizing profits in the long-run?

3.7.1 Definition of churn

A customer is deemed as churned if they go 30 days without completing a paid trip 30 days after their last successful trip.

3.7.2 Data Cleaning

Removing of customers with less than 2 months of tenure as at 30 days into the measurement point: This allows the model to only looks at customers with at least 60 days of tenure; translating to 30 days of customer activity/inactivity forming the independent features and 30 days to determine customer churn, making the dependent variable

Removing outlier customers*:* System bugs will typically manifest as customers having a very high frequency of engagement with products/services offered and/or high monetary values in relatively short periods of time. These instances are dropped from the data

For record, feature values that lie above $Q1 + 1.5 * IQR$ results in the record being dropped.

Where

i. Q1 = 25th percentile
ii. IQR = interquartile range

3.7.3 Feature Engineering

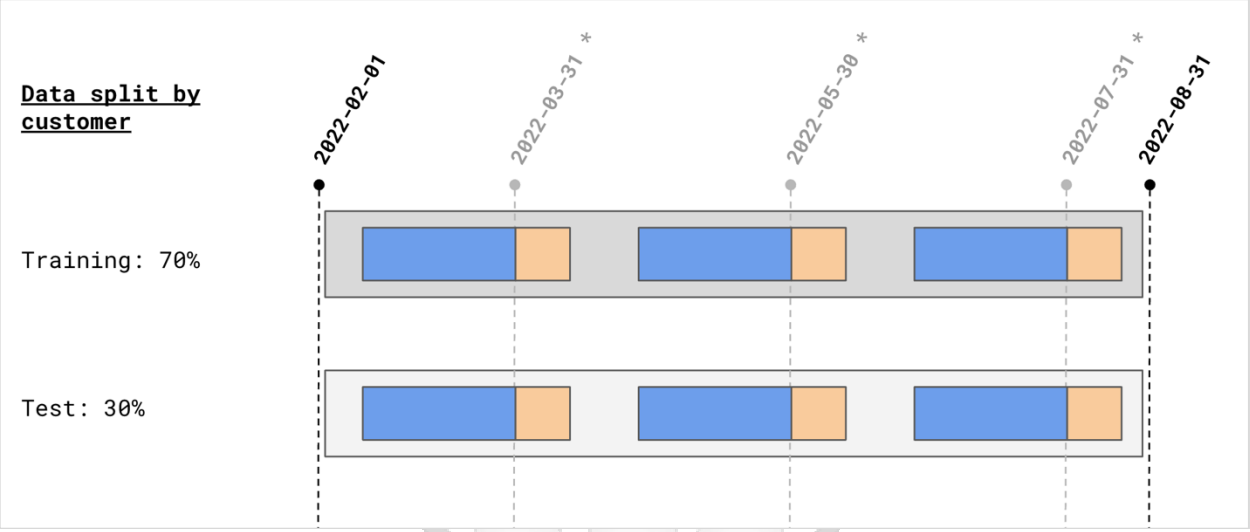

Figure 3.10:Training and test data split showing sliding window sampling

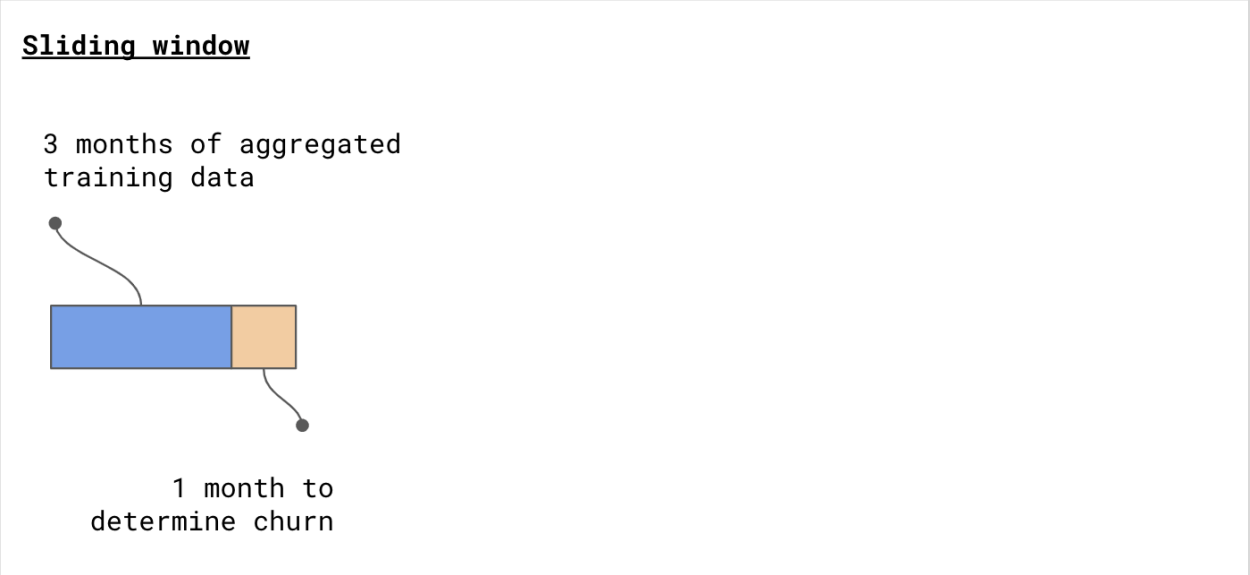

Figure 3.11: Sliding window legend

Selection of measurement point(s) in time (2022-03-31, 2022-05-30, 2022-07-31)

Selection of measurement points creates an effect similar to a sliding window over time series data. At a particular point, customer activity data is collected and aggregated at customer level in the 90 days leading up to it; over the subsequent 30 days a customer is deemed to have churned if they do not have a bike ride.

With the three points selected, a customer with tenure covering the entire period will have a maximum of 3 records; with 3 pairs of predictor variables (aggregated at customer level over 90 days) and target variable (churned or not determined over 30 days)

Table 3.8: Sample data schema

user_id	measurement_date	X_1	X_2	…	X_n	y
1	2022-05-01					1
2	2022-06-01					0
3	2022-07-01					1
…						
n	2022-05-01					0

3.7.3.1 Normalization of customer activity metrics to weekly level

Training data is prepared from the raw data; to avoid and minimize the chances of a concept drift the prediction model looks at a max minimum of 3 months of customer activity to predict customer churn in the next 30 days.

Metrics to measure customer activity is normalized to a weekly level to account for new and older customers. For instance, a customer with 30 days of tenure at the measurement point with 10 completed rides will have a similar weekly ride count as a customer with 60 days of tenure with 20 completed rides.

Table 3.9: Model independent variables

Variable	Description	Type	Calculation
user_id	Unique customer identifier	Profile	
age	As at measurement point	Profile	
days_since_last_activity	As at measurement point	All engagement	

days_since_last_bike_ride	No. of day from last ride	Ride engagement	
active_days_per_week	No. of active days per week	All engagement	
active_bike_days_per_week	No. of ride-active days per week	Ride engagement	
bike_count_per_week	No. of rides per week	Ride engagement	
bike_cash_count_per_week	No. of rides paid in cash per week	Ride engagement	
bike_cashless_count_per_week	No. of rides paid in cashless per week	Ride engagement	
send_count_per_week	No. of parcel trips in a week	Parcel engagement	
topup_count_per_week	No. of wallet topups in a week	Telco service engagement	
airtime_count_per_week	No. of airtime purchases in a week	Telco service engagement	
data_count_per_week	No. data purchases in a week	Telco service engagement	
bill_count_per_week	No. of bill payments in a week	Fintech engagement	
p2v_count_per_week	No. of passenger to vendor payments	Fintech engagement	
o2p_count_per_week	No. of office to passenger disbursements	Fintech engagement	
s2p_count_per_week	No. of SafeBoda driver to passenger transactions	Fintech engagement	
p2p_count_per_week	No. of passenger to passenger transactions	Fintech engagement	
p2a_count_per_week	No. of passenger to agent transactions	Fintech engagement	
a2p_count_per_week	No. of agent to passenger transactions	Fintech engagement	
p2b_count_per_week	No. of passenger to bank transactions	Fintech engagement	sum(*activity_metric) / (measurement_date - min(date_joined, measurement_period)) / 7
p2mm_count_per_week	No. of passenger to mobile money	Fintech engagement	

	transactions	
mobile_network_count_per_week	No. of mobile money transactions	Fintech engagement
fintech_count_per_week	No. of all fintech transactions	Fintech engagement
all_activity_count_per_week	No. of all activity on the platform	All engagement
bike_value_per_week	Value of bike rides	Ride engagement
bike_cash_value_per_week	Value of bike rides paid in cash	Ride engagement
bike_cashless_value_per_week	Value of bike raid paid in cashless	Ride engagement
send_value_per_week	Value of parcel trips	Ride engagement
topup_value_per_week	Value of topups to the SafeBoda wallet	Fintech engagement
airtime_value_per_week	Value of artime purchase from wallet	Fintech engagement
data_value_per_week	Value of data purchased from wallet	Fintech engagement
bill_value_per_week	Value of bills paid from wallet	Fintech engagement
p2v_value_per_week	Value of passenger to vendor transactions	Fintech engagement
o2p_value_per_week	Value of office to passenger transactions	Fintech engagement
s2p_value_per_week	Valie fo driver to passenger transactions	Fintech engagement
p2p_value_per_week	Value of passenger to passenger transactions	Fintech engagement
p2a_value_per_week	Valie of passenger to agent transactions	Fintech engagement
a2p_value_per_week	Value of agent to passenger transactions	Fintech engagement
p2b_value_per_week	Value of passenger to bank transactions	Fintech engagement

p2mm_value_per_week	Value of passenger to mobile money transactions	Fintech engagement	
mobile_network_value_per_week	Value of all mobile network transactions	Fintech engagement	
fintech_value_per_week	Value of all fintech transactions	Fintech engagement	
price_per_km	Average in the activity period	Ride engagement	
trip_rating	Average in the activity period	Ride engagement	
perc_cancelled_requests	Proportion of requests that were cancelled cancelled_requests / total_requests	Ride engagement	
perc_completed_requests	Proportion of requests that lead to successful trips completed_trips / total_requests	Ride engagement	
perc_failed_requests	Proportion of requests that were not matched to a driver failed_requests / total_requests	Ride engagement	

Table 3.10: Model target variable

Variable	Data Type	Description
is_churned	Int64	Binary value, 1 if the user churned in the 30-day evaluation period; 0 otherwise

Customer retention in a multi-service platform like SafeBoda hinges on how different types of user behaviours and profile characteristics signal ongoing engagement or impending churn. Profile variables such as age (customer tenure) provide baseline demographic context for risk stratification. Engagement metrics grouped into overall activity, ride usage, parcel trips, telco services, and fintech transactions map directly onto established retention frameworks like RFM (Recency–Frequency–Monetary) and platform-specific loyalty theories. By explaining each cluster of variables in relation to behavioural economics, lifecycle marketing, and domain

intuition, we clarify why they are predictive of churn and how they can guide targeted interventions.

Customer age (tenure on the platform) offers demographic anchors for understanding baseline churn risk. Tenure provides a proxy for habit formation: customers who have been active longer are more likely to have integrated the app into their daily routines, reducing their propensity to churn (Becerril-Castrillejo & Muñoz-Gallego, 2022). All-platform engagement metrics such as days since last activity, active days per week, and all activity count per week correspond to the Recency and Frequency dimensions of RFM models, where more recent and frequent interactions signal stronger habit formation and reduced churn likelihood. High overall activity indicates diverse feature use and a greater "sunk cost" in the platform, making departure less attractive.

Ride engagement metrics like days since last bike ride, bike count per week, and price per kilometer capture core service utilization. Frequent and recent ride usage fosters service dependency, and higher average trip values often reflect greater willingness to pay; both are linked to lower churn in ride-hailing contexts (Nguyen-Phuoc, Diep, & Lester, 2020). Furthermore, positive trip rating and low percentage of failed requests enhance satisfaction and trust, which are critical for retention (Sasiprapha, Khahan, & Kaptun, 2025). Send count per week and send value per week metrics tap into the parcel-delivery arm of the business. Research in logistics and delivery services shows that reliable and frequent parcel experiences build loyalty, as customers view the platform as a one-stop solution for mobility and goods transport (Lai, Hyunmi, & Mingjie, 2022).

Topup count per week, airtime count per week, and data count per week reflect usage of integrated telecom services. In super-app models, cross-selling telco offerings enhances user "stickiness," because telecommunications are daily necessities and regularly recurring transactions reinforce habitual engagement (Gao, de Haan, Iguácel, & Sese, 2023). Fintech engagement variables such as bill count per week, p2p count per week, and the aggregate fintech count per week, along with their corresponding monetary values, align with financial inclusion and loyalty research indicating that consumers who adopt in-app payment features develop

higher switching costs and exhibit greater inertia (Basri, Iqbal , & Naveen, 2022). The regular use of fintech services, especially wallet top-ups and peer-to-peer payments—creates network effects that further entrench users in the ecosystem.

By grouping variables according to the type of engagement and linking each group to theoretical and empirical retention frameworks such as RFM, behavioural economics of habit formation, and platform-specific loyalty models, we ensure that every predictor in the churn model has a clear conceptual rationale.

3.7.3.2 Checking and treating for class imbalances (retained vs churned customers)

An analysis of the distribution of the target variable showed that the data was imbalanced, with a higher proportion of the records recording the customers has having retained (is_churned = 0) than those that churned in the measurement period (is_churned = 1).

Class imbalances hinder the performance of standard classification algorithms (Japkowicz, 2000) and is especially true for cases of rare events such as fraud detection in network systems where instances of breaches or attached are extremely low compared to what is considered normal traffic. For this dataset random over sampling was employed dues to its relatively low-cost, as measured by the performance of the classifiers compared to over-sampling techniques (Drummond, 2003)

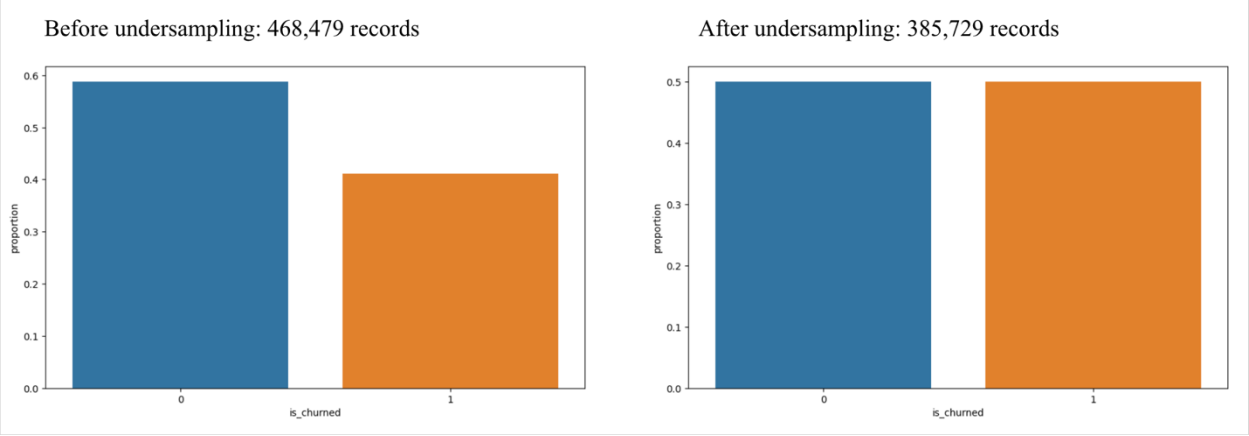

Figure 3.12: Distribution of churned vs retained customer before and after under sampling

3.7.3.3 Eliminating highly correlated independent features

Pearson's correlation co-efficient was calculated on each pair of explanatory variables and; single feature from pairs of features showing strong correlation between each other is was eliminated. This reduces model complexity and chances of overfitting.

Table 3.11: List of highly correlated features eliminated

Feature
fintech_count_per_week
bike_value_per_week
mobile_network_count_per_week
bike_cashless_value_per_week
send_value_per_week
bike_count_per_week
days_since_last_bike_ride
bike_cash_value_per_week
active_bike_days_per_week
fintech_count_per_week
bike_value_per_week

3.7.4 Feature Selection

3.7.4.1 Pearson Correlation

Pearson's correlation co-efficient is used for each feature against the target variable. Features with high correlation to the target variable are selected by ranking from highest to lowest and picking the top 20 variables

$$r = \frac{\sum(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum(X_i - \bar{X})^2 \sum(Y_i - \bar{Y})^2}}$$

where:

i. X_i and Y_i are the individual data points for variables X and Y.

ii. $\bar{X}$ and $\bar{Y}$ are the means of X and Y, respectively.

3.7.4.2 Mutual Information

Mutual information (MI) quantifies the amount of information obtained about one random variable, through another random variable, thus determining the relevance of each feature with respect to the target variable.

$$I(X;Y) = \sum_{x \in X} \sum_{y \in Y} P(x,y) \log\left(\frac{P(x,y)}{P(x)P(y)}\right)$$

Where:

i. $I(X;Y)$ is the mutual information between variables I (feature) and $\boldsymbol{Y}$ (target).
ii. $P(x,y)$ is the joint probability distribution of X and Y.
iii. $P(x)$ and $P(y)$ are the marginal probability distributions of X and Y.

MI was calculated for each feature against the target variable and then ranked based on MI scores, higher being better. The top 20 feature were then selected

3.7.4.3 Recursive Feature Elimination (RFE) with Logit Estimator

This employs logistic regression to model the relationship between the independent variables (features) and a binary dependent variable (target) and provides support of the significance of features in predicting the target outcome.

$$P(Y = 1|X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n)}}$$

Where:

i. $P(Y = 1 \mid X)$ is the probability of the positive class given the feature set XXX.
ii. β_0 is the intercept.
iii. $\beta_1, \beta_2, \ldots, \beta_n$ are the coefficients for features $X_1, X_2, \ldots, X_n$
iv. e is the base of the natural logarithm.

Features were selected based on the significance of their coefficients by evaluating the p-values associated with each coefficient. Features with high p-values (> 0.05) are sequentially remove until the desired count of feature is reached.

3.7.4.4 Recursive Feature Elimination (RFE) with XGBoost

This method iteratively employs XGBoost, an implementation of gradient boosting for classification, fitting the model against the data and eliminates the least important features based on feature importance scores until the set number of features is reached, in this case 20.

3.7.4.5 Embedded Methods - Logistic Regression

A regularized logistic regression model was fit on the data and features with non-zero co-efficients selected. Lasso regularization is employed to select a subset of variables that are most important in predicting the outcome variable. The lasso method adds a penalty term to the regression coefficients to shrink them towards zero. This has the effect of reducing the impact of less important variables and potentially setting their coefficients to zero (Fonti & Belister, 2017)

3.7.4.6 Embedded Methods – Random Forest

A random forest classification model was fit on the data and importance of each feature was evaluated based on Gini impurity in the decision trees; important features contribute to reducing impurity than lest important ones. The top 20 features are selected.

3.7.4.7 Embedded Methods – Light Gradient Boosting

Light Gradient Boosting computes feature importance based on the contribution of each feature to the reduction in the loss function over all trees in the model during the training step. Top 20 features that significantly contribute to improving model accuracy are considered more important and are selected.

3.7.5 Combined Feature Selection by Voting Mechanism

Each feature selection model computes support value for each feature. In this case ‘true’ or ‘false’ value for support is assigned to each feature. A vote of 1 is assigned to ‘true’ values and 0 for ‘false values.

Each feature selection models effectively votes for each feature and a tally is computed from the votes of all models. The top 20 features by votes are selected as inputs for the classification models.

VT OMNES
VNVM SINT

Table 3.12: Feature selection votes by model

#	feature	Pearson	Mutual Information	RFE-XGB	RFE-Logistic Reg.	Embedded - Logistic Reg.	Embedded - Random Forest	Embedded Light GBM	Total Votes
1	active_days_per_week	1	1	1	1	1	1	1	7
2	bike_cash_count_per_week	1	1	1	1	1	1	1	7
3	price_per_km	1	1	1	1	1	1	1	7
4	days_since_last_activity	1	1	1	1	1	1	1	7
5	all_activity_count_per_week	1	1	1	1	1	1	1	7
6	age	1	1	1	0	0	1	1	5
7	s2p_count_per_week	1	1	1	1	1	0	0	5
8	perc_failed_requests	1	1	1	0	0	1	1	5
9	perc_completed_requests	1	1	1	0	0	1	1	5
10	perc_cancelled_requests	1	1	1	0	0	1	1	5
11	topup_value_per_week	1	1	1	1	0	0	1	5
12	trip_rating	1	1	1	0	0	1	1	5
13	bike_cashless_count_per_w	1	1	1	1	1	0	0	5
14	airtime_count_per_week	1	0	0	1	1	0	0	4
15	bill_count_per_week	1	1	0	1	0	0	0	3
16	topup_count_per_week	0	1	1	1	1	0	0	3
17	airtime_value_per_week	1	0	0	1	1	0	0	3
18	o2p_count_per_week	1	0	0	1	1	0	0	3
19	p2a_count_per_week	1	0	0	1	1	0	0	3
20	a2p_count_per_week	0	1	1	1	0	0	0	3
21	o2p_value_per_week	0	0	0	1	0	0	0	2
22	data_count_per_week	0	1	1	0	0	0	0	2
23	p2p_count_per_week	0	1	0	1	0	0	0	2
24	data_value_per_week	0	0	1	1	0	0	0	2
25	s2p_value_per_week	0	1	1	0	0	0	0	2
26	p2mm_value_per_week	1	0	0	0	0	0	0	2
27	send_count_per_week	0	0	1	1	0	0	0	1
28	p2mm_count_per_week	0	0	1	1	0	0	0	1
29	p2p_value_per_week	0	1	0	0	0	0	0	1
30	p2v_count_per_week	1	0	0	0	0	0	0	1
31	p2a_value_per_week	0	0	0	0	0	0	0	0
32	p2b_value_per_week	0	0	0	0	0	0	0	0
33	mobile_network_value_per_	0	0	0	0	0	0	0	0
34	fintech_value_per_week	0	0	0	0	0	0	0	0
35	p2v_value_per_week	0	0	0	0	0	0	0	0
36	bill_value_per_week	0	0	0	0	0	0	0	0
37	p2b_count_per_week	0	0	0	0	0	0	0	0
38	a2p_value_per_week	0	0	0	0	0	0	0	0

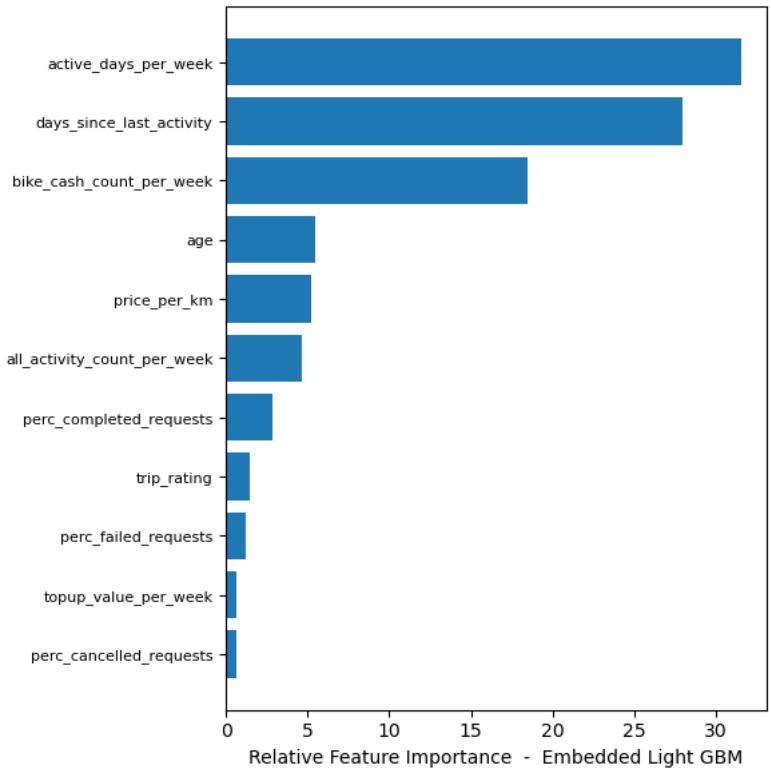

Figure 3.13: Sample relative feature importance (Embeded Light GBM)

3.7.6 Feature Scaling

Exploratory data analysis reveals that features are of varying magnitudes and are highly skewed. Scaling ensures that features with larger values do not exhibit dominance especially for algorithms that are sensitive to distance like k-nearest neighbours and support vector machines. Gradient decent models that move in steps towards minimums will also converge faster (Jadhav, Dhaulakhandi, Shandilya, Malviya, & Mewada, 2023).

For this dataset, the Yeo-Johnson transformation is used

$Case\ 1: \boldsymbol{x \geq 0}$

$$T(x, \lambda) = \begin{cases} \frac{(x+1)^{\lambda}-1}{\lambda}, & \text{if } \lambda \neq 0 \\ \ln(x+1), & \text{if } \lambda = 0 \end{cases}$$

$Case\ 2: \boldsymbol{x < 0}$

$$T(x, \lambda) = \begin{cases} \frac{-(-x+1)^{2-\lambda}+1}{2-\lambda}, & \text{if } \lambda \neq 2 \\ -\ln(-x+1), & \text{if } \lambda = 2 \end{cases}$$

Here:

i. $T(x, \lambda)$ is the transformed value of x

ii. λ is a parameter estimated by the transformer to maximize the log-likelihood of a normal distribution. The transformation finds the optimal λ using maximum likelihood estimation (MLE) to make the transformed data as close to Gaussian as possible.

iii. The data is standardized to have zero mean and unit variance.

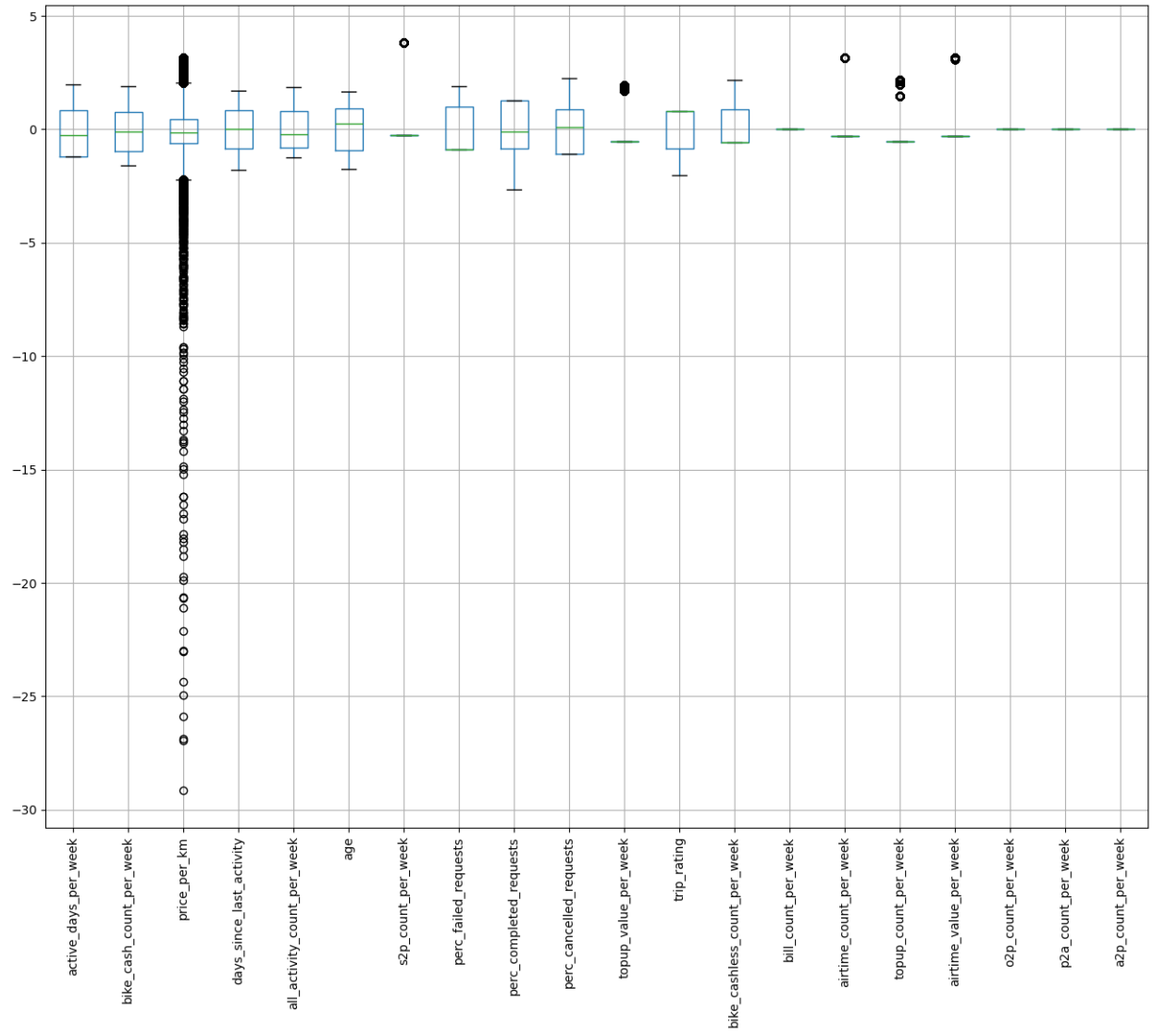

Figure 3.14: Box-plot of transformed and scaled features

3.7.7 Fitting Classifier Models

The process of fitting machine learning models to data is central to this research. The goal is to assess the performance of various classifiers on the given dataset to identify the best model for predicting customer churn.

The Gradient Boosting Classifier was chosen for its ability to sequentially fit new decision trees to the residual errors of prior models, thereby capturing complex non-linear patterns and interactions in the churn data. The XGBoost Classifier was selected as an optimized implementation of gradient boosting that incorporates regularization to prevent overfitting and

leverages second-order derivatives for faster convergence. Logistic Regression was included as a baseline linear model due to its interpretability and efficiency in estimating probabilities for binary outcomes through maximum-likelihood estimation.

The Decision Tree Classifier was employed for its intuitive, rule-based splits that facilitate straightforward interpretation of key churn drivers. K-Neighbors Classifier was applied as a non-parametric method that classifies customers based on the majority label among their k most similar peers in feature space.

Random Forest was included to reduce variance through an ensemble of decorrelated trees, improving generalization by averaging across multiple subsampled models. The Stochastic Gradient Descent (SGD) Classifier was selected for its computational efficiency on large datasets, updating model weights incrementally using random subsets of data. It enables efficient training on massive datasets and supports online learning, all while accommodating a variety of convex loss functions and regularization schemes. These properties make SGD classifier particularly suitable for large-scale churn prediction tasks, where speed, continual adaptation, and model flexibility are paramount.

The Linear Support Vector Classifier (SVC) was chosen for its robustness in finding the maximum-margin hyperplane that separates churned from retained users in high-dimensional space. Finally, the Multi-Layer Perceptron (MLP) Classifier was included due to its capacity to model complex, non-linear relationships among behavioral features via multiple hidden layers and backpropagation learning. In many real-world tabular settings, simpler MLPs have been shown to outperform more complex deep-learning architectures on heterogeneous datasets, because they avoid over-parameterization and focus representational power where it matters most. Recent studies also demonstrate that MLP-based ensembles can match or exceed the accuracy of advanced specialized networks while requiring fewer compute resources (Chernov, 2024)

i. **Gradient Boosting Classifier**: Gradient Boosting is an ensemble technique that builds a model sequentially by fitting new models to the residual errors of previous ones. This model is particularly effective for capturing complex patterns and interactions in data. The Gradient Boosting Classifier is fitted by iteratively adding decision trees to minimize

the loss function, typically using a mean squared error or log loss criterion. The number of trees, learning rate, and tree depth are key factors considered during training.

ii. **XGBoost Classifier**: XGBoost is an optimized implementation of gradient boosting that incorporates regularization to prevent overfitting. Like Gradient Boosting, it builds decision trees in a sequential manner. The fitting process involves selecting the optimal hyperparameters, including the number of estimators, learning rate, maximum depth, and subsampling rate. The training process uses the gradient boosting algorithm with second-order derivatives to speed up the learning process.

iii. **Logistic Regression Classifier**: Logistic Regression is a simple yet powerful linear model for binary classification. The model is fitted by finding the coefficients (weights) that maximize the likelihood of the observed data, using a logistic function to predict probabilities. The fitting process involves solving an optimization problem using gradient descent or other optimization algorithms, depending on the solver chosen (e.g., 'liblinear' or 'saga'). The regularization strength, represented by the parameter C, is tuned to prevent overfitting.

iv. **Decision Trees Classifier**: Decision Trees work by recursively splitting the data into subsets based on feature values, aiming to create partitions that maximize the homogeneity of the target variable within each subset. The fitting process involves selecting the best features and split points at each node using a splitting criterion such as Gini impurity or entropy. The algorithm continues until a stopping criterion is met (e.g., tree depth, minimum samples per leaf). Overfitting is controlled by tuning hyperparameters like tree depth and minimum samples per leaf.

v. **K-Neighbors Classifier**: The K-Neighbors Classifier is a non-parametric, lazy learning algorithm that assigns a class label based on the majority class among the k-nearest neighbors of a data point. During the fitting process, the model stores the training data and computes the distance between a query point and all other points. The number of neighbors (k) and the distance metric (e.g., Euclidean, Manhattan) are key parameters selected to ensure model efficiency and accuracy.

vi. **Random Forest Classifier**: Random Forest is an ensemble learning technique that builds multiple decision trees and combines their predictions. Each tree is trained on a random subset of features and data points, helping reduce variance and increase generalization.

The fitting process involves training each tree using a subset of the data, with the number of estimators (trees) and maximum depth being key tuning parameters. Random Forest can handle both regression and classification tasks, and the trees are aggregated using majority voting for classification.

vii. **Stochastic Gradient Descent (SGD) Classifier**: The SGD Classifier is a linear model optimized using stochastic gradient descent. The model is fitted by iteratively updating the coefficients based on random subsets of the training data. Each update moves the coefficients in the direction of the gradient of the loss function. The learning rate, regularization penalty (L1, L2), and number of iterations are key parameters considered in the fitting process. SGD is computationally efficient for large datasets.

viii. **Linear Support Vector Classifier**: The Linear Support Vector Classifier (SVC) is based on the concept of finding a hyperplane that best separates the classes in the feature space. The model is fitted by maximizing the margin between the classes while minimizing classification errors. The key parameter is the penalty parameter C, which controls the trade-off between achieving a wide margin and minimizing classification errors. The model is trained using optimization techniques like quadratic programming.

ix. **Multi-layer Perceptron (MLP) Classifier**: The MLP Classifier is a type of artificial neural network with one or more hidden layers. Each neuron in a layer is connected to neurons in the adjacent layers, and the network learns through backpropagation by adjusting weights based on the error between predicted and actual values. The fitting process involves choosing the number of layers, the number of neurons per layer, the activation function, and the learning rate. The network is trained using a gradient descent-based optimizer, which updates the weights iteratively.

3.7.8 Model Hyper-parameter tuning

Model performance heavily depends on the proper selection of hyperparameters, which are parameters set before the learning process begins. Hyperparameter tuning aims to identify the optimal set of hyperparameters that enhances model performance on a given task. The following classifiers are being applied in this study, and hyperparameter tuning is conducted for each model to ensure the best results:

i. **Gradient Boosting Classifier**: This model builds an ensemble of decision trees sequentially, where each tree tries to correct the errors of the previous one. Key hyperparameters include the learning rate, the number of estimators (trees), and the maximum depth of each tree. To tune these parameters, GridSearchCV is employed to find the optimal balance between bias and variance.

ii. **XGBoost Classifier**: XGBoost is a gradient boosting method that is highly efficient. The key hyperparameters for tuning include the learning rate, maximum depth, subsample ratio, and the number of estimators. Regularization parameters (lambda and alpha) also play a crucial role in controlling overfitting. The optimization of these hyperparameters is done using GridSearchCV, which exhaustively searches for the best combination of parameters.

iii. **Logistic Regression Classifier**: Logistic regression, a linear model for binary classification, has hyperparameters such as regularization strength (C) and the solver (e.g., 'liblinear', 'saga'). These are tuned to avoid underfitting or overfitting. Cross-validation is used to evaluate the model for each set of parameters.

iv. **Decision Trees Classifier**: Decision trees are highly interpretable but prone to overfitting. Key hyperparameters like the maximum depth of the tree, minimum samples per leaf, and criterion (Gini or entropy) are tuned to enhance model generalization. GridSearchCV is used to search over a range of parameter values.

v. **K-Neighbors Classifier**: This model relies on the number of neighbors to classify data points. Tuning the number of neighbors (k) and the distance metric (Euclidean, Manhattan, etc.) is essential. A range of values for k is tested using cross-validation, and the best-performing k is selected.

vi. **Random Forest Classifier**: Random forests are ensembles of decision trees. Important hyperparameters include the number of trees (n_estimators), maximum depth, and the number of features considered for splitting nodes (max_features). GridSearchCV is employed to optimize these hyperparameters.

vii. **Stochastic Gradient Descent (SGD) Classifier**: This linear classifier uses gradient descent for optimization. Hyperparameters like the learning rate, penalty type (L2 or L1), and maximum number of iterations are tuned. A grid search over a range of values helps determine the best set of parameters.

viii. **Linear Support Vector Classifier**: This linear model is tuned based on the penalty parameter C and the kernel type. The choice of kernel (linear, polynomial, radial basis function) is considered, and cross-validation is used to evaluate the performance of each combination.

ix. **Multi-layer Perceptron Classifier**: An artificial neural network-based model, where key hyperparameters include the number of hidden layers, the number of neurons in each layer, and the activation function. Grid search is used for a comprehensive search of these parameters to optimize model performance.

For all models, **cross-validation** is performed to evaluate the performance of each set of hyperparameters. The goal is to identify the optimal hyperparameters that reduce bias, prevent overfitting, and improve the generalization of each classifier. The grid search and random search methods are used as appropriate for each model to ensure the best possible outcomes while avoiding the computational expense of exhaustive search.

By tuning hyperparameters, we ensure that each model operates at its highest potential, producing accurate and reliable predictions while maintaining generalizability to unseen data.

3.7.9 Classifier Model Evaluation

Model performance is evaluated using a set of standard metrics that provide a comprehensive understanding of how well each classifier performs on both training and testing data. These metrics are especially important when dealing with imbalanced datasets, as they allow for a more nuanced evaluation beyond simple accuracy. The following metrics and techniques are used to assess the performance of the classifiers:

3.7.9.1 Confusion Matrix

The confusion matrix is a fundamental tool for evaluating classification models. It provides a detailed breakdown of the predicted versus actual class labels, making it possible to calculate the number of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). The matrix allows for a deeper understanding of how well the model is distinguishing between classes, especially when dealing with imbalanced data.

3.7.9.2 Accuracy

Accuracy is the most straightforward metric, measuring the proportion of correct predictions (both true positives and true negatives) among the total number of observations. It is calculated as:

$$accuracy = (TP + TN) / total\ observations$$

3.7.9.3 Precision

Precision measures the proportion of positive predictions that were actually correct. It is an important metric when the cost of false positives is high. For example, in medical diagnoses, a false positive may lead to unnecessary treatments or interventions.

Precision is calculated as:

$$precision = TP / (TP + FP)$$

This metric helps to evaluate how reliable the model's positive predictions are.

3.7.9.4 Recall (Sensitivity)

Recall, also known as sensitivity, measures the model's ability to identify all relevant positive cases. It is particularly useful when the cost of false negatives is high, such as in fraud detection or disease diagnosis, where failing to identify a positive case can have serious consequences. Recall is calculated as:

$$recall = TP \,/\, (TP + FN)$$

A high recall means that the model is effectively capturing most of the positive instances, but it may come at the cost of lower precision (more false positives).

3.7.9.5 F1 Score

The F1 score provides a balanced evaluation metric by combining precision and recall into a single value, making it particularly useful for imbalanced datasets where both false positives and false negatives are important. The F1 score is the harmonic mean of precision and recall, calculated as:

$$F1\ score = 2 \cdot (precision * recall) \,/\, (precision + recall)$$

This metric provides a robust evaluation when there is a need to balance the trade-off between precision and recall.

3.7.9.6 ROC Curve (Receiver Operating Characteristic Curve) and AUC (Area Under the Curve)

The ROC curve is a graphical representation of a classifier's performance across all classification thresholds. It plots the true positive rate (recall) against the false positive rate (1 - specificity). The ROC curve illustrates how well the model discriminates between the positive and negative classes. The area under the ROC curve (AUC) provides a scalar value that summarizes the overall ability of the model to distinguish between the classes. AUC values range from 0 to 1, with 1 indicating perfect classification performance.

Evaluation Process

All models are evaluated on both the training and test datasets to assess their ability to generalize to new, unseen data. The metrics are calculated for each model based on the predictions made on the test set after training.

In addition to these individual metrics, cross-validation is used to further assess the stability and reliability of the model performance across different subsets of the data. This helps to prevent overfitting and ensures that the model's performance is not overly dependent on a specific data split.

By using a combination of these metrics, this research ensures that each model is rigorously evaluated across multiple dimensions, providing a comprehensive picture of its strengths and weaknesses. The results are then analyzed to identify the best-performing classifier based on the specific requirements of the problem, such as handling class imbalance, minimizing false positives or false negatives, and ensuring reliable predictions.

Chapter 4: Results and Analysis

4.1 Model Results

4.1.1 Gradient Boosting Classifier

Table 4.1: Gradient boosting classifier - model evaluation on train data

Train					
Accuracy	0.7363218536				
Confusion Matrix	0	1			
0	92,332	42,633			
1	28,563	106,483			
	0	1	accuracy	macro avg	weighted avg
precision	0.7637371273	0.7140950669	0.7363218536	0.7389160971	0.7389086511
recall	0.6841181047	0.7884942908	0.7363218536	0.7363061978	0.7363218536
f1-score	0.7217384507	0.7494527769	0.7363218536	0.7355956138	0.7355997708
support	134965	135046	0.7363218536	270011	270011

Table 4.2: Gradient boosting classifier - model evaluation on test data

Test					
Accuracy	0.7367415895				
Confusion Matrix	0	1			
0	39,620	18,280			
1	12,184	45,635			
	0	1	accuracy	macro avg	weighted avg
precision	0.7648058065	0.7139951498	0.7367415895	0.7394004782	0.7394182612
recall	0.6842832470	0.7892734222	0.7367415895	0.7367783346	0.7367415895
f1-score	0.7223072996	0.7497494537	0.7367415895	0.7360283767	0.7360187723
support	57900	57819	0.7367415895	115719	115719

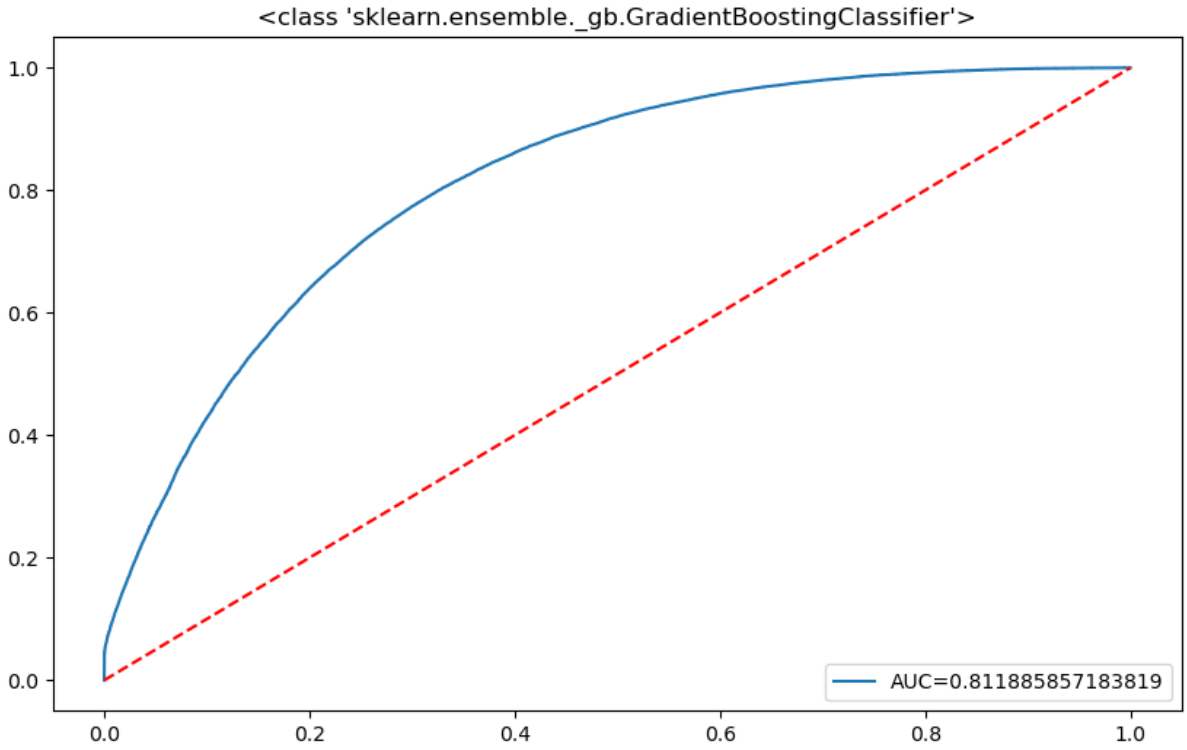

Figure 4.1: Gradient boosting classifier - receiver operating curve

On the test set (Table 4.2), the model correctly identifies 39,620 retained customers (true negatives) but misclassifies 18,280 as churners; yielding a false positive rate of 31.6 % (18,280 / 57,900). It also misses 12,184 actual churners; producing a false negative rate of 21.1 % (12,184 / 57,819). From a cost perspective, each false negative represents a lost lifetime value: customers who slip through without intervention often require substantially higher incentives later (or may never return), while each false positive leads to unnecessary retention spend on loyal users, eroding ROI on targeted campaigns.

4.1.2 XGBoost Classifier

Table 4.3: XGBoost classifier - model evaluation on train data

Train					
Accuracy	0.7490398539				
Confusion Matrix	0	1			
0	94,335	40,630			
1	27,132	107,914			
	0	1	accuracy	macro avg	weighted avg
precision	0.7766306898	0.7264783498	0.7490398539	0.7515545198	0.7515469973
recall	0.6989589894	0.7990906802	0.7490398539	0.7490248348	0.7490398539
f1-score	0.7357506083	0.7610564547	0.7490398539	0.7484035315	0.7484073273
support	134965	135046	0.7490398539	270011	270011

Table 4.4: XGBoost classifier - model evaluation on test data

Test					
Accuracy	0.7360589013				
Confusion Matrix	0	1			
0	39,674	18,226			
1	12,317	45,502			
	0	1	accuracy	macro avg	weighted avg
precision	0.7630936124	0.7140032639	0.7360589013	0.7385484381	0.738565619
recall	0.6852158895	0.7869731403	0.7360589013	0.7360945149	0.7360589013
f1-score	0.7220609513	0.748714489	0.7360589013	0.7353877202	0.7353783918
support	57900	57819	0.7360589013	115719	115719

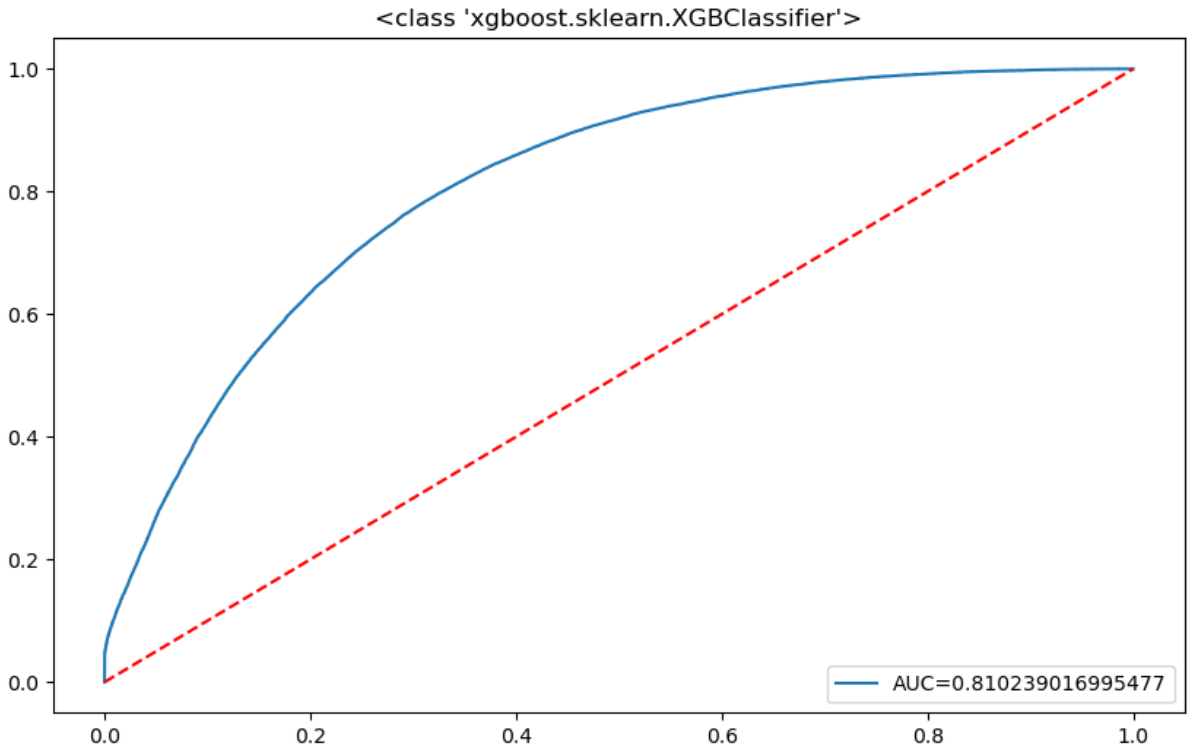

Figure 4.2: XGBoost classifier - receiver operating curve

In Table 4.4, XGBoost records 39,674 true negatives and 45,502 true positives but commits 18,226 false positives (31.5 %) and 12,317 false negatives (21.3 %). The near-identical error profile to Gradient Boosting suggests similar trade-offs: over-targeting stable customers inflates marketing costs, whereas under-detecting churners risks unmitigated revenue loss and higher future reacquisition expenses.

4.1.3 Logistic Regression Classifier

Table 4.5:Logistic regression classifier: model evaluation on training data

Train					
Accuracy	0.7297924899				
Confusion Matrix	0	1			
0	96,039	38,926			
1	34,033	101,013			
	0	**1**	**accuracy**	**macro avg**	**weighted avg**
precision	0.7383526047	0.7218359428	0.7297924899	0.7300942738	0.7300917964
recall	0.7115844849	0.7479895739	0.7297924899	0.7297870294	0.7297924899
f1-score	0.724721454	0.7346800735	0.7297924899	0.7297007637	0.7297022575
support	134965	135046	0.7297924899	270011	270011

Table 4.6: Logistic regression classifier - model evaluation on test data

Test					
Accuracy	0.7298974239				
Confusion Matrix	0	1			
0	41,202	16,698			
1	14,558	43,261			
	0	**1**	**accuracy**	**macro avg**	**weighted avg**
precision	0.7389167862	0.7215096983	0.7298974239	0.7302132423	0.7302193345
recall	0.7116062176	0.7482142548	0.7298974239	0.7299102362	0.7298974239
f1-score	0.7250043991	0.7346193686	0.7298974239	0.7298118839	0.7298085188
support	57900	57819	0.7298974239	115719	115719

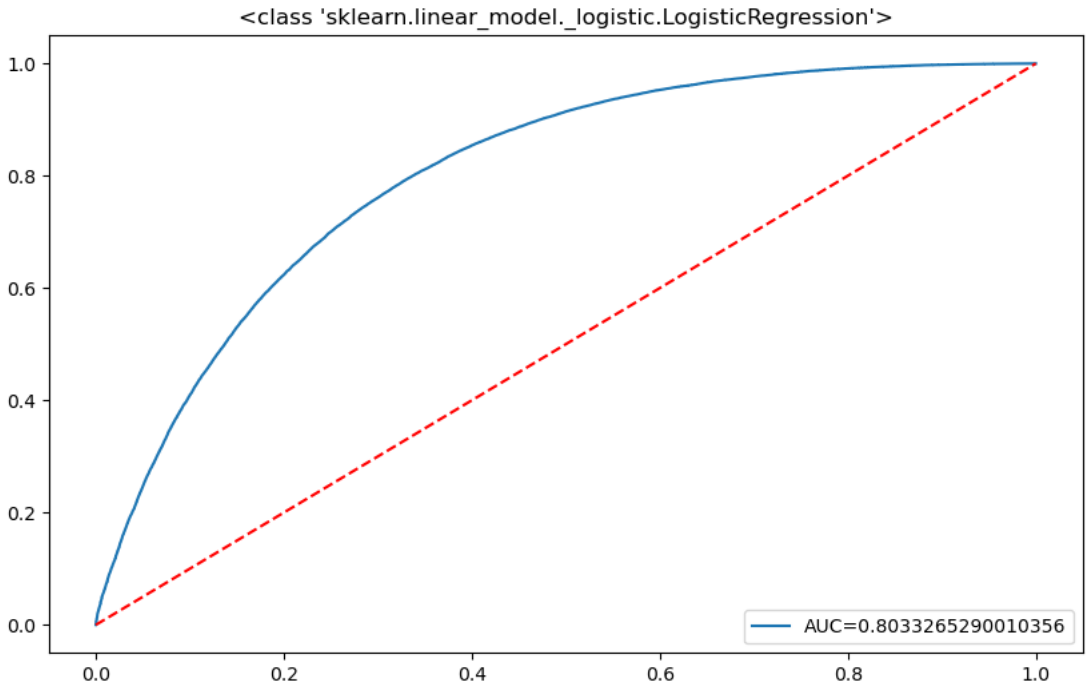

Figure 4.3: Logistic regression classifier - receiver operating curve

For Logistic Regression (Table 4.6), the false positive rate drops to 28.8 % (16,698 / 57,900) but false negatives climb to 25.2 % (14,558 / 57,819). Here, cost savings on wasted retention outreach are modestly improved, yet the higher churn blind-spot means more customers churn unnoticed, potentially translating into outsized revenue leakage if high-value customers are misclassified.

4.1.4 Decision Trees Classifier

Table 4.7: Decision trees classifier - model evaluation on train data

Train					
Accuracy	0.9999777787				
Confusion Matrix	**0**	**1**			
0	134,964	1			
1	5	135,041			
	0	**1**	**accuracy**	**macro avg**	**weighted avg**
precision	0.9999629545	0.9999925949	0.9999777787	0.9999777747	0.9999777791
recall	0.9999925907	0.9999629756	0.9999777787	0.9999777831	0.9999777787
f1-score	0.9999777723	0.999977785	0.9999777787	0.9999777787	0.9999777787
support	134965	135046	0.9999777787	270011	270011

Table 4.8: Decision trees classifier - model evaluation on test data

Test					
Accuracy	0.6432392261				
Confusion Matrix	**0**	**1**			
0	37,431	20,469			
1	20,815	37,004			
	0	**1**	**accuracy**	**macro avg**	**weighted avg**
precision	0.6426364042	0.6438501557	0.6432392261	0.64324328	0.6432428552
recall	0.6464766839	0.6399972327	0.6432392261	0.6432369583	0.6432392261
f1-score	0.644550824	0.6419179128	0.6432392261	0.6432343684	0.6432352899
support	57900	57819	0.6432392261	115719	115719

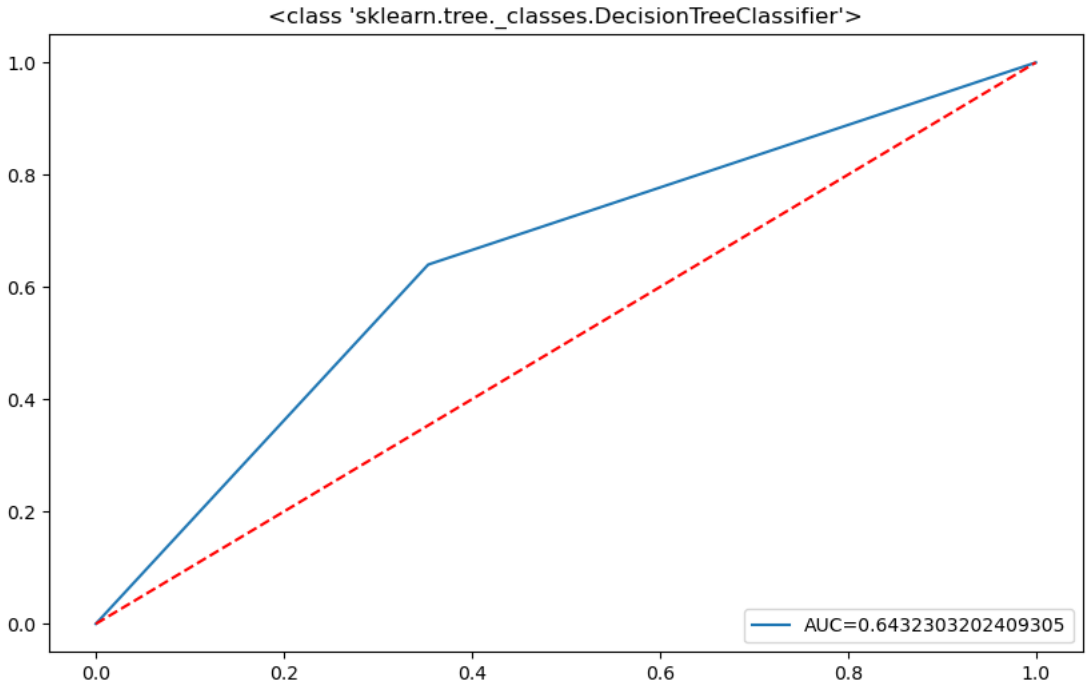

Figure 4.4: Decision trees classifier - receiver operating curve

The Decision Tree (Table 4.8) exhibits a 35.4 % false positive rate (20,469 / 57,900) and a 36.0 % false negative rate (20,815 / 57,819). Such elevated misclassification implies substantial costs: frequent unnecessary offers to retained users, and worse, many churners slip through; indicating that this model's simplicity may undermine practical retention budgeting and failure to prevent attrition.

4.1.5 KNeighbors Classifier

Table 4.9: KNeighbors classifier - model evaluation on train data

Train					
Accuracy	0.7882641818				
Confusion Matrix	**0**	**1**			
0	103,850	31,115			
1	26,056	108,990			

	0	1	accuracy	macro avg	weighted avg
precision	0.799424199	0.7779165626	0.7882641818	0.7886703808	0.7886671548
recall	0.7694587486	0.8070583357	0.7882641818	0.7882585421	0.7882641818
f1-score	0.7841553058	0.7922195449	0.7882641818	0.7881874253	0.7881886349
support	134965	135046	0.7882641818	270011	270011

Table 4.10: KNeighbours classifier - model evaluation on test data

Test					
Accuracy	0.6962210182				
Confusion Matrix	**0**	**1**			
0	39,287	18,613			
1	16,540	41,279			
	0	**1**	**accuracy**	**macro avg**	**weighted avg**
precision	0.703727587	0.6892239364	0.6962210182	0.6964757617	0.6964808378
recall	0.6785319516	0.7139348657	0.6962210182	0.6962334087	0.6962210182
f1-score	0.690900138	0.7013618099	0.6962210182	0.696130974	0.6961273125
support	57900	57819	0.6962210182	115719	115719

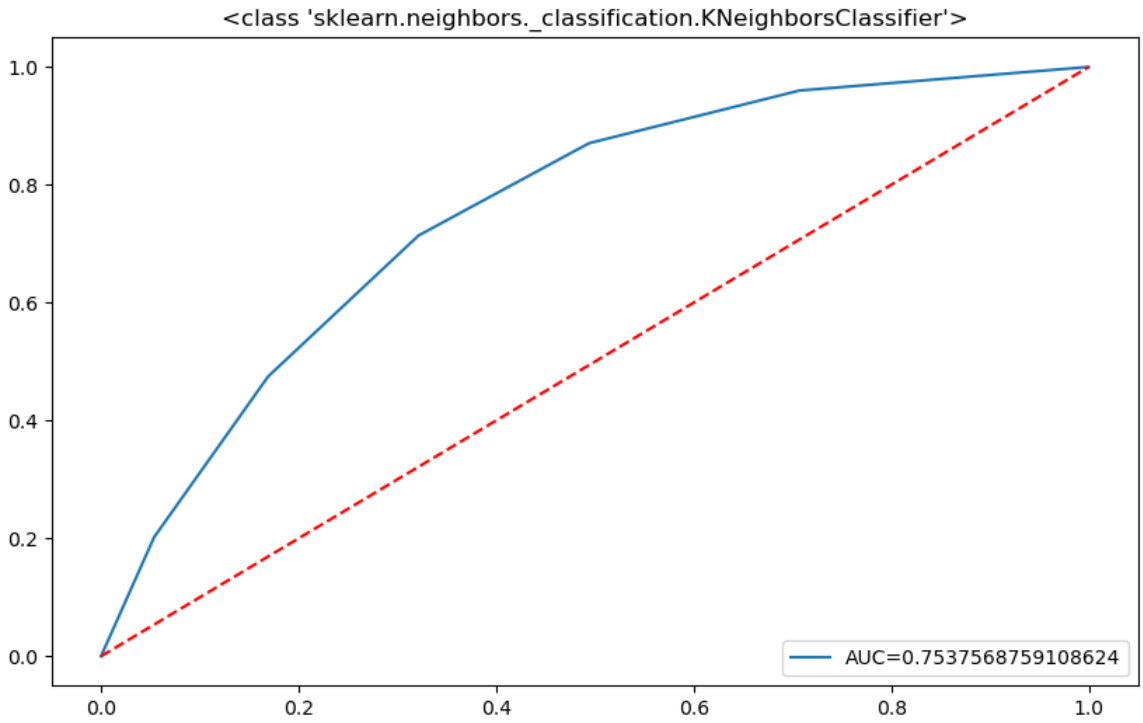

Figure 4.5: KNeighbours classifier - receiver operating curve

K-Neighbors (Table 4.10) mislabels 32.1 % of retained customers (18,613 / 57,900) and misses 28.6 % of churners (16,540 / 57,819). While closer to the ensemble models, each false positive remains a misallocated incentive, and each false negative represents foregone revenue; underscoring the need to balance K's value for interpretability against these tangible budgetary impacts.

4.1.6 Random Forest Classifier

Table 4.11: Random forest classifier - model evaluation on train data

Train					
Accuracy	0.9994703919				
Confusion Matrix	**0**	**1**			
0	134,913	52			
1	91	134,955			
	0	**1**	**accuracy**	**macro avg**	**weighted avg**
precision	0.9993259459	0.9996148348	0.9994703919	0.9994703903	0.9994704337
recall	0.9996147149	0.9993261555	0.9994703919	0.9994704352	0.9994703919
f1-score	0.9994703096	0.9994704743	0.9994703919	0.9994703919	0.999470392
support	134965	135046	0.9994703919	270011	270011

Table 4.12: Random forest classifier - model evaluation on test data

Test					
Accuracy	0.7148091498				
Confusion Matrix	**0**	**1**			
0	40,432	17,468			
1	15,534	42,285			
	0	**1**	**accuracy**	**macro avg**	**weighted avg**
precision	0.7224386234	0.7076632136	0.7148091498	0.7150509185	0.7150560897
recall	0.6983074266	0.7313339906	0.7148091498	0.7148207086	0.7148091498
f1-score	0.7101680923	0.7193039159	0.7148091498	0.7147360041	0.7147328067
support	57900	57819	0.7148091498	115719	115719

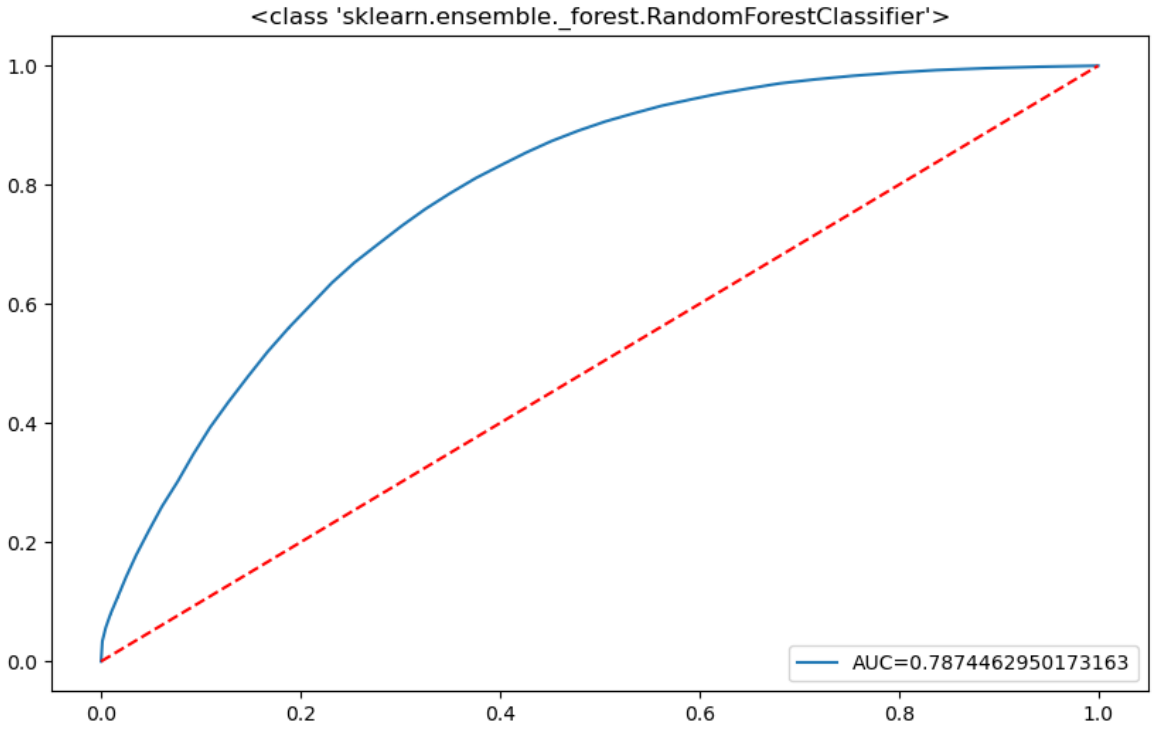

Figure 4.6: Random forest classifier - receiver operating curve

On test data (Table 4.12), Random Forest yields a 30.2 % false positive rate (17,468 / 57,900) and a 26.9 % false negative rate (15,534 / 57,819). By reducing both error types relative to simpler trees, this model slightly lowers wasted outreach and avoids more churn, yet each remaining misclassification still carries quantifiable costs in promotional spend and lost customer lifetime value. Notably, the confusion matrices for the training and test data bear a stark difference; very low values for false positive and false negative rates in the test data indicate model overfitting

4.1.7 Stochastic Gradient Decent Classifier

Table 4.13: Stochastic gradient classifier - model evaluation on train data

Train					
Accuracy					
Confusion Matrix	**0**	**1**			
0	97,803	37,162			
1	35,995	99,051			
	0	**1**	**accuracy**	**macro avg**	**weighted avg**
precision	0.7309750519	0.7271772885	0.7290591865	0.7290761702	0.7290756006
recall	0.7246545401	0.733461191	0.7290591865	0.7290578655	0.7290591865
f1-score	0.7278010738	0.7303057226	0.7290591865	0.7290533982	0.7290537739
support	134965	135046	0.7290591865	270011	270011

Table 4.14: Stochastic gradient classifier - model evaluation on test data

Test					
Accuracy	0.7283246485				
Confusion Matrix	**0**	**1**			
0	41,881	16,019			
1	15,419	42,400			
	0	**1**	**accuracy**	**macro avg**	**weighted avg**
precision	0.7309075044	0.7257912665	0.7283246485	0.7283493855	0.7283511761
recall	0.7233333333	0.7333229561	0.7283246485	0.7283281447	0.7283246485
f1-score	0.7271006944	0.7295376727	0.7283246485	0.7283191836	0.7283183307
support	57900	57819	0.7283246485	115719	115719

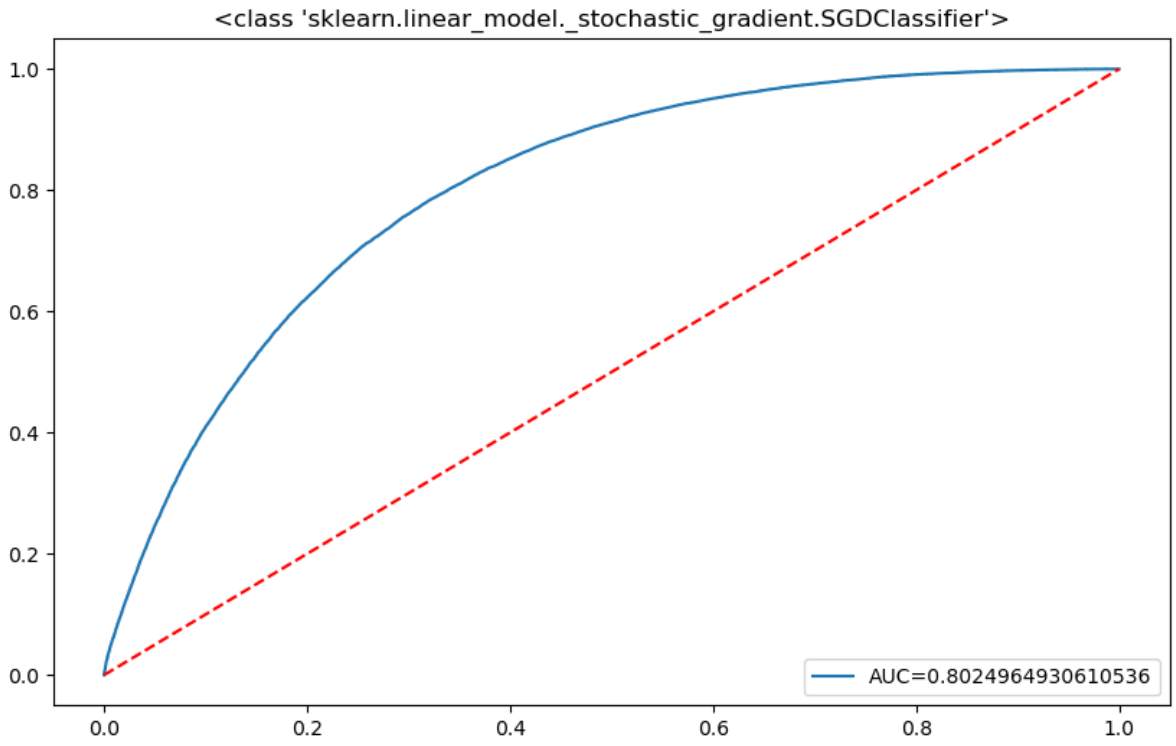

Figure 4.7: Stochastic gradient descent classifier - receiver operating curve

SGD's test confusion matrix (Table 4.14) shows 16,019 false positives (27.7 %) and 15,419 false negatives (26.6 %). Its efficient incremental training comes with modest misclassification trade-offs: reduced over-spend on loyal customers but elevated unaddressed churn that may erode long-term profitability if high-churn segments go unnoticed.

4.1.8 Linear Support Vector Classifier

Table 4.15: Linear support vector classifier - model evaluation on train data

Train					
Accuracy	0.7299517427				
Confusion Matrix	**0**	**1**			
0	95,304	39,661			
1	33,255	101,791			

	0	1	accuracy	macro avg	weighted avg
precision	0.7413249947	0.7196151345	0.7299517427	0.7304700646	0.7304668083
recall	0.7061386285	0.7537505739	0.7299517427	0.7299446012	0.7299517427
f1-score	0.7233041393	0.7362874234	0.7299517427	0.7297957813	0.7297977288
support	134965	135046	0.7299517427	270011	270011

Table 4.16: Linear support vector classifier - model evaluation on train data

Test					
Accuracy	0.7302517305				
Confusion Matrix	**0**	**1**			
0	40,901	16,999			
1	14,216	43,603			
	0	**1**	**accuracy**	**macro avg**	**weighted avg**
precision	0.7420759475	0.7194977063	0.7302517305	0.7307868269	0.730794729
recall	0.7064075993	0.7541292655	0.7302517305	0.7302684324	0.7302517305
f1-score	0.7238026138	0.7364065495	0.7302517305	0.7301045816	0.7301001704
support	57900	57819	0.7302517305	115719	115719

The Linear SVC (Table 4.16) commits 16,999 false positives (29.3 %) alongside 14,216 false negatives (24.6 %). While providing robust margin-based separation, the model still requires careful calibration: each false positive inflates retention costs, and each false negative signals revenue at risk and potentially higher re-engagement expenses.

4.1.9 Multi-layer Perceptron Classifier

Table 4.17: Multi-layer perceptron (MLP) classifier - model evaluation on train data

Train					
Accuracy	0.7301406239				
Confusion Matrix	**0**	**1**			
0	100,758	34,207			
1	38,658	96,388			

	0	**1**	**accuracy**	**macro avg**	**weighted avg**
precision	0.722714753	0.7380680731	0.7301406239	0.730391413	0.7303937159
recall	0.7465491053	0.7137419842	0.7301406239	0.7301455448	0.7301406239
f1-score	0.7344386091	0.7257012283	0.7301406239	0.7300699187	0.7300686082
support	134965	135046	0.7301406239	270011	270011

Table 4.18: Multi-layer perceptron (MLP) classifier - model evaluation on test data

Test					
Accuracy	0.7301221061				
Confusion Matrix	**0**	**1**			
0	43,120	14,780			
1	16,450	41,369			
	0	**1**	**accuracy**	**macro avg**	**weighted avg**
precision	0.7238542891	0.7367718036	0.7301221061	0.7303130463	0.7303085254
recall	0.7447322971	0.7154914474	0.7301221061	0.7301118723	0.7301221061
f1-score	0.7341448881	0.7259757125	0.7301221061	0.7300603003	0.7300631594
support	57900	57819	0.7301221061	115719	115719

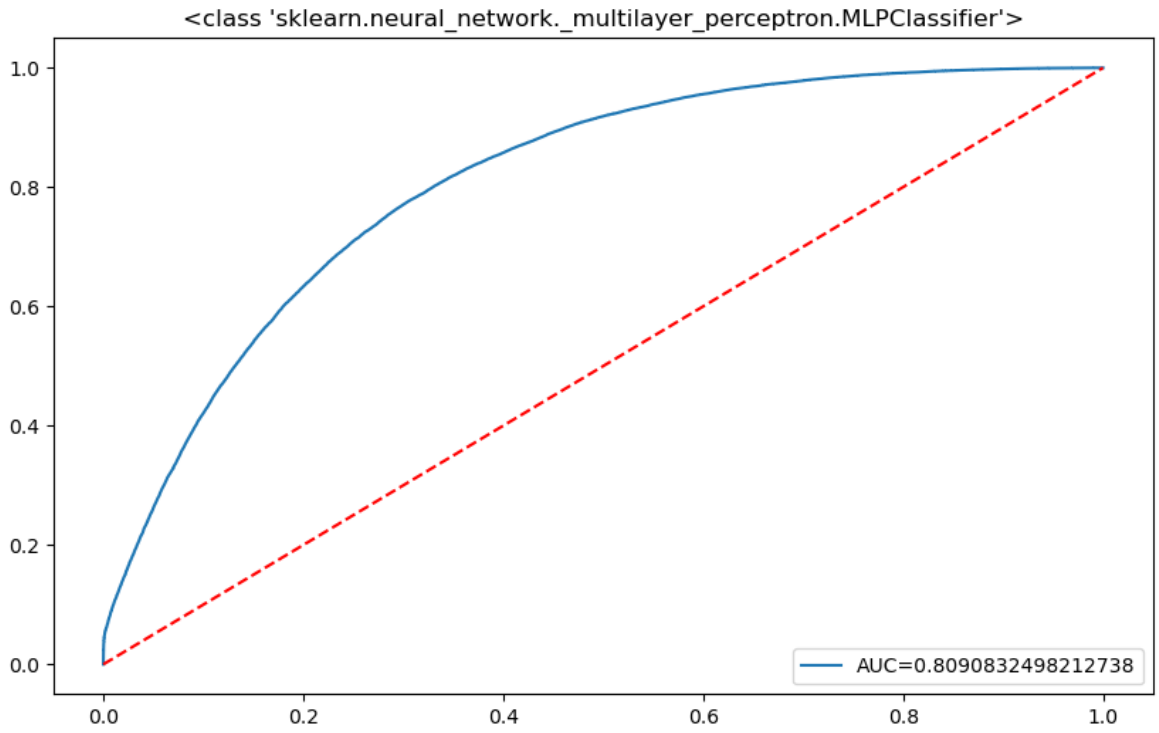

Figure 4.8: Multi-layer perceptron classifier - receiver operating curve

The MLP (Table 4.18) achieves its lowest false positive rate of 25.5 % (14,780 / 57,900) but incurs a higher false negative rate of 28.5 % (16,450 / 57,819). This skew suggests that while fewer loyal customers receive unnecessary retention offers, a larger share of at-risk users remain undetected; implying potential revenue loss that may outweigh savings on outreach unless mitigated by subsequent intervention strategies.

4.2 Model Evaluation

The following table presents the performance metrics for each model evaluated on the dataset. The metrics include Accuracy, AUC (Area Under the Curve), Precision, Recall, F1 Score, and a Total Score that aggregates these measures.

i. **Gradient Boosting Classifier** performs the best overall, with an accuracy of 73.67%, an AUC of 0.8119, and strong precision (0.7140), recall (0.7893), and F1 score (0.7497). The model's Total Score of 3.8016 reflects its ability to balance all evaluation metrics effectively.

ii. **XGBoost Classifier** follows closely behind, with an accuracy of 73.61%, an AUC of 0.8102, and similar precision (0.7140), recall (0.7870), and F1 score (0.7487). The Total Score of 3.7960 is nearly identical to that of Gradient Boosting, indicating that XGBoost performs similarly but slightly less efficiently.

iii. **Logistic Regression** provides competitive results with an accuracy of 72.99% and an AUC of 0.8033. While its precision (0.7215) and recall (0.7482) are strong, its F1 score of 0.7346 results in a Total Score of 3.7376, and ranks behind the ensemble models.

iv. **Stochastic Gradient Descent (SGD)** achieves an accuracy of 72.83%, an AUC of 0.8025, and solid precision (0.7258), recall (0.7333), and F1 score (0.7295). Its Total Score of 3.7195 indicates a competitive performance, though marginally lower than Logistic Regression.

v. **Multi-layer Perceptron (MLP)** has an accuracy of 73.01%, an AUC of 0.8091, and its best performance in precision (0.7368). However, its recall (0.7155) and F1 score (0.7260) are slightly lower, resulting in a Total Score of 3.7174, indicating that while it is effective, its overall performance is just shy of the top performers.

vi. **Linear SVC** shows an accuracy of 73.03%, an AUC of 0.7997, and good recall (0.7541). However, its Total Score of 3.6949 is slightly lower than the MLP and Logistic Regression due to a slightly lower F1 score (0.7364) and precision (0.7195).

vii. **Random Forest** performs slightly worse with an accuracy of 71.48% and an AUC of 0.7874. While its precision (0.7077) and recall (0.7313) are reasonable, its F1 score (0.7193) results in a lower Total Score of 3.6606.

viii. **K-Neighbors** shows the lowest accuracy among the classifiers at 69.62%, with an AUC of 0.7538. Despite having decent recall (0.7139) and precision (0.6892), its relatively

lower F1 score (0.7014) leads to a Total Score of 3.5545, indicating that it underperforms compared to other models.

ix. **Decision Tree** performs the weakest overall, with an accuracy of 64.32%, an AUC of 0.6432, and a Total Score of 3.2122. Despite relatively balanced values for precision (0.6439) and recall (0.6400), the Decision Tree's overall performance is hindered by lower values across the board.

Conclusion

The results indicate that **Gradient Boosting** and **XGBoost** are the top-performing classifiers, with very similar metrics across all performance indicators. These ensemble methods show the best ability to balance accuracy, precision, recall, and F1 score, making them the most reliable models for this task. **Logistic Regression** and **SGD** also show strong performance, while models like **Random Forest**, **K-Neighbors**, and **Decision Tree** exhibit comparatively lower performance, especially in terms of accuracy and F1 score. The **Decision Tree** exhibits the lowest performance across all metrics.

Table 4.19: Model evaluation on test data

Model	Accuracy	AUC	Precision	Recall	F1 Score	Total Score
Gradient Boosting	0.7367	0.8119	0.7140	0.7893	0.7497	3.8016
XGBoost	0.7361	0.8102	0.7140	0.7870	0.7487	3.7960
Logistic Regression	0.7299	0.8033	0.7215	0.7482	0.7346	3.7376
Stochastic Gradient Descent	0.7283	0.8025	0.7258	0.7333	0.7295	3.7195
Multi-layer Perceptron	0.7301	0.8091	0.7368	0.7155	0.7260	3.7174
Linear SVC	0.7303	0.7997	0.7195	0.7541	0.7364	3.6949
Random Forest	0.7148	0.7874	0.7077	0.7313	0.7193	3.6606
KNeighbors	0.6962	0.7538	0.6892	0.7139	0.7014	3.5545
Decision Tree	0.6432	0.6432	0.6439	0.6400	0.6419	3.2122

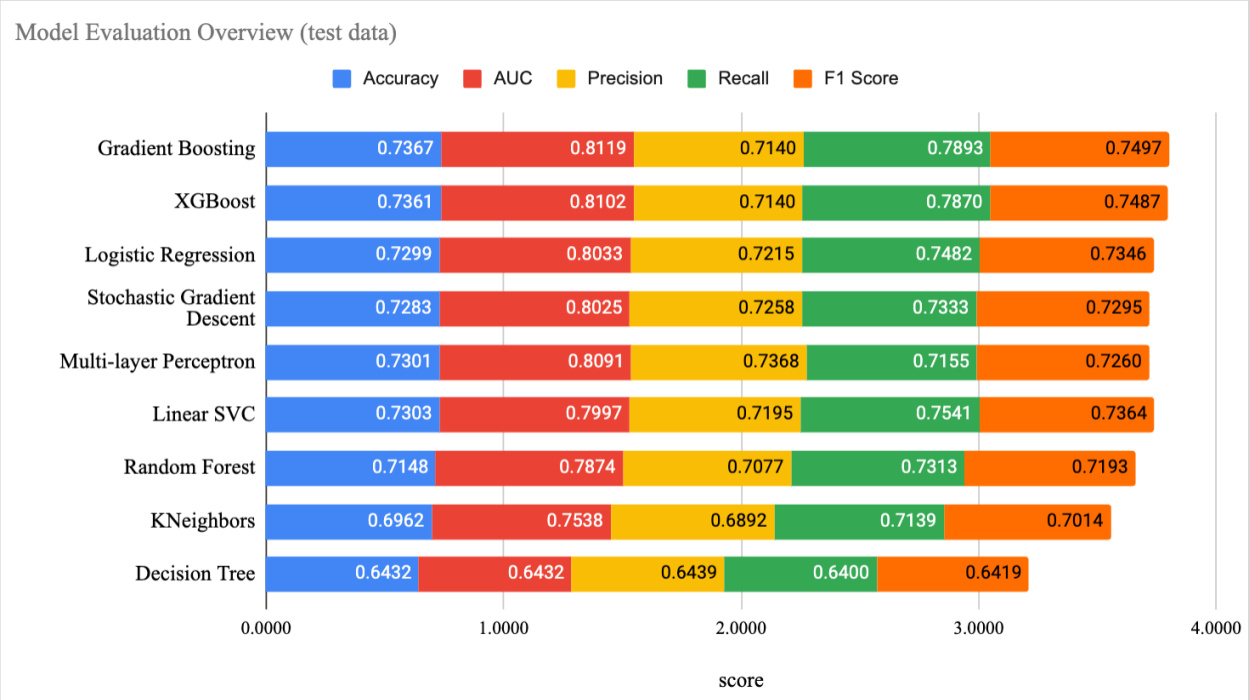

Figure 4.9: Model evaluation on test data

4.3 Hyperparameter Tuning

Following the initial model evaluation, hyperparameter tuning was conducted on the highest-performing models, namely Gradient Boosting, XGBoost, Logistic Regression, Stochastic Gradient Descent (SGD), and Multi-layer Perceptron (MLP). The results from the hyperparameter optimization process indicate improved performance across the models, with slight variations in accuracy and AUC values between training and testing datasets.

Table 4.20: Accuracy and AUC metrics of tuned models

Model	Training Accuracy %	Testing Accuracy %	Test AUC
Tuned Gradient Boosting	74.135869	73.747613	0.812331
Tuned XGBoost	73.710701	73.662061	0.812015
Tuned Logistic Regression	72.990730	73.011346	0.803388
Tuned Stochastic Gradient Decent	74.926947	73.505647	0.809959
Tuned Multi-layer Perceptron	73.644777	73.594656	0.810607

Tuned Gradient Boosting

The **Tuned Gradient Boosting** model achieved a training accuracy of 74.14% and a testing accuracy of 73.75%. The testing AUC for this model is 0.8123, indicating a very slight improvement from the pre-tuning performance (AUC = 0.8119). The small increase in testing accuracy demonstrates that hyperparameter tuning helped refine the model slightly, but the overall performance remains robust. Gradient Boosting continues to be one of the top-performing models, maintaining strong generalization capabilities.

Tuned XGBoost

The **Tuned XGBoost** model showed a training accuracy of 73.71% and a testing accuracy of 73.66%, with a testing AUC of 0.8120. Like Gradient Boosting, XGBoost's performance showed minimal improvement after tuning. The model’s testing accuracy is nearly identical to the pre-tuned version (73.61%), and the AUC remains comparable (AUC = 0.8102), confirming that XGBoost is a highly stable model. The slight improvements observed post-tuning reflect enhanced fine-tuning but show that the model's capacity to generalize to unseen data is already strong.

Tuned Logistic Regression

The **Tuned Logistic Regression** model exhibited a training accuracy of 72.99% and a testing accuracy of 73.01%. The testing AUC is 0.8034, which is a slight improvement from the pre-tuning AUC (0.8033). Although Logistic Regression showed some minor improvements in performance after hyperparameter tuning, its overall results are still slightly behind the ensemble methods like Gradient Boosting and XGBoost. However, it remains a strong contender for simpler models that require less computational power.

Tuned Stochastic Gradient Descent (SGD)

The **Tuned Stochastic Gradient Descent** model showed the highest training accuracy among the tuned models at 74.93%, with a testing accuracy of 73.51%. The testing AUC of 0.8099 is slightly lower than the AUC of the ensemble methods but still competitive. The significant increase in training accuracy suggests that the model became more specialized to the training data after tuning. However, this increase did not translate into a large improvement in testing accuracy, indicating that SGD's performance after tuning is robust but not superior to models like Gradient Boosting.

Tuned Multi-layer Perceptron (MLP)

The **Tuned Multi-layer Perceptron** model showed a training accuracy of 73.64% and a testing accuracy of 73.59%. The testing AUC is 0.8106, which is a small improvement from the pre-tuning AUC (0.8091). While the MLP demonstrated improved performance after hyperparameter tuning, it still did not outperform the top models, Gradient Boosting and XGBoost.

Summary of Findings

Overall, the results from the hyperparameter tuning process show that **Gradient Boosting** and **XGBoost** continue to perform at the top level, with minimal improvements in accuracy and AUC following the tuning process. Both models exhibit strong generalization abilities, reflected in their similar performance on both training and testing datasets.

Stochastic Gradient Descent (SGD) showed a significant increase in training accuracy but did not result in a large improvement in testing accuracy, suggesting the model may be slightly

overfitted to the training data. Despite this, it remains competitive with other models in terms of testing accuracy and AUC.

Multi-layer Perceptron (MLP) also experienced some improvements after hyperparameter tuning, though its performance remains slightly behind the ensemble methods in terms of testing accuracy and AUC.

Logistic Regression, while showing minor improvements, still lags behind the other models in terms of accuracy and AUC. However, its simpler structure and fast training time make it a practical choice for problems requiring quick solutions with acceptable performance.

In conclusion, **Gradient Boosting** and **XGBoost** remain the top-performing models, while **Stochastic Gradient Descent** and **Multi-layer Perceptron** offer competitive results. The hyperparameter tuning process has refined these models, enhancing their performance without major changes to their relative rankings.

4.4 Final Model Selection

In this study, **Tuned XGBoost** has been selected over **Tuned Gradient Boosting** for the final model choice due to its superior performance on larger datasets and its computational efficiency. While both models perform similarly in terms of accuracy and AUC, there are several reasons why XGBoost is more suitable for this research.

Performance on Larger Datasets

XGBoost is known for its high performance on large datasets, particularly when there are a large number of features and complex interactions. Several studies have demonstrated that XGBoost often outperforms traditional gradient boosting models, especially as the size and complexity of the data increase. This is because XGBoost implements optimizations like gradient-based optimization, parallelization, and tree pruning, which enable it to scale better and achieve faster convergence than standard Gradient Boosting methods.

For instance, (Chen & Guestrin, 2016), in their seminal paper on XGBoost, highlight its superior efficiency and speed, particularly when dealing with large-scale data. They emphasize that XGBoost's ability to handle sparse data, its regularization techniques, and parallelized tree

construction contribute significantly to its improved performance, especially when the dataset grows in size or complexity. For the data set in this study XGboost takes 6 seconds to train, compared to 46 seconds with Gradient boosting. These advantages make XGBoost a preferable choice for problems that require fast training times and high model accuracy on large datasets.

Regularization and Overfitting

XGBoost incorporates regularization (L1 and L2), which helps prevent overfitting by penalizing overly complex models. This is particularly beneficial when working with large datasets that may contain noise or irrelevant features. Gradient Boosting, while effective, lacks the regularization capabilities of XGBoost, which can lead to a higher risk of overfitting, especially on large and high-dimensional datasets.

Computational Efficiency

XGBoost is widely recognized for its computational efficiency compared to standard Gradient Boosting models. By leveraging parallel computation and distributed computing, XGBoost significantly reduces training time. Moreover, XGBoost's ability to perform early stopping allows the algorithm to halt training when further improvement in model performance is negligible, optimizing both time and computational resources.

Hyperparameter Tuning Flexibility

XGBoost offers a wide range of hyperparameters that can be tuned, providing greater flexibility in fine-tuning the model to the specific requirements of the dataset. This flexibility, combined with the model's robust optimization methods, enables it to achieve better performance compared to Gradient Boosting when applied to complex or large-scale problems. In the current research, the fine-tuned XGBoost model showed comparable, results on the testing set, solidifying its choice for the final model.

4.5 Explaining the Model

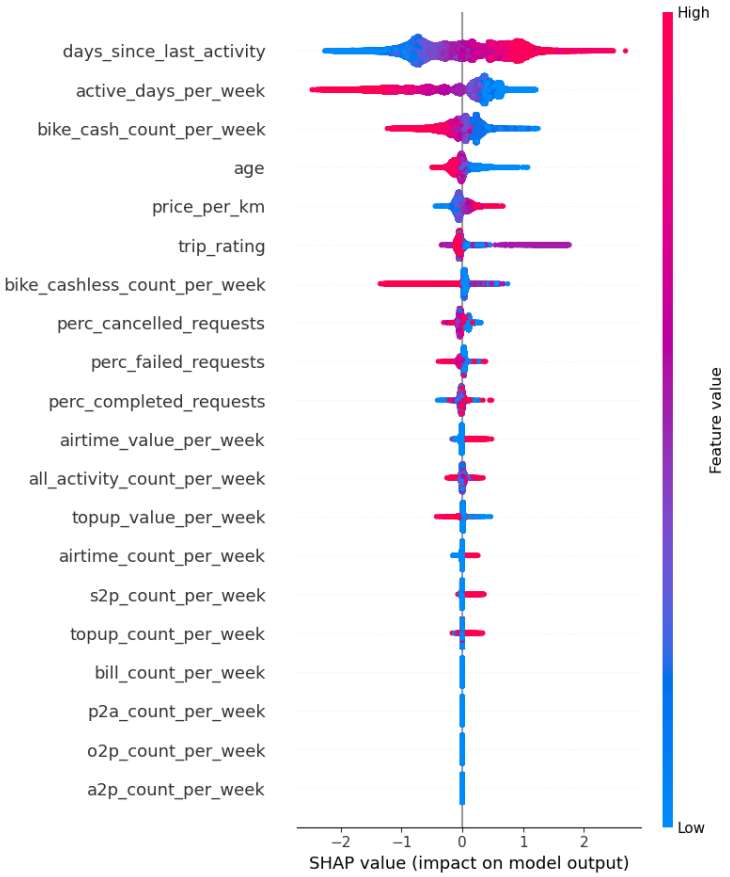

Figure 4.10: SHAP summary plot with feature values

Overall, the plot shows that the model's predictions are primarily influenced by:

i. **days_since_last_activity:** This feature has the largest positive impact on the model's output. High values of this feature tend to push the model's prediction towards the "postivie" end, meaning customer is likely to churn.

ii. **bike_cash_count_per_week:** This feature also has a significant positive impact. Customers who frequently use cash payments for bike rides are more likely to be classified with a higher risk of churning. This is in contrast with **bike_cashless_count_per_week** which tend to predict customer retention

iii. **active_days_per_week:** Customers with a higher number of active days per week are more likely to be classified as having a lower likelihood of churning.

iv. **trip_rating:** Lower trip ratings are associated with higher risk classifications.

Other notable features and their impact:

i. **active_days_per_week:** Customers with a higher number of active days per week are more likely to be classified as "High" risk.

ii. **price_per_km:** Higher prices per kilometer tend to push predictions towards "High" risk.

iii. **perc_cancelled_requests, perc_failed_requests, perc_completed_requests:** These features related to request outcomes have a relatively small impact on the model's predictions.

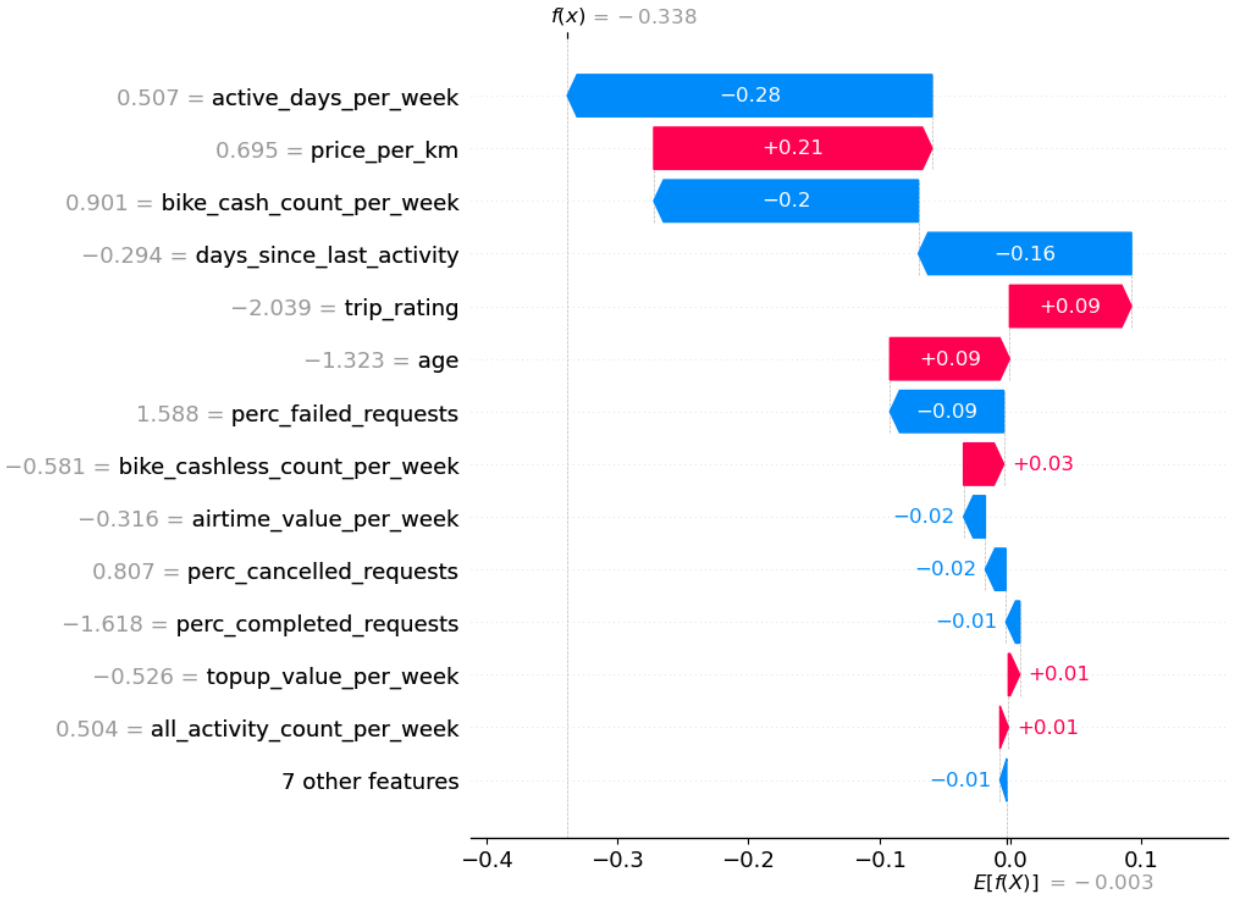

Figure 4.11: SHAP waterfall for a sample prediction instance

Base Value (E[f(X)]): The base value of -0.003 indicates that, on average, the model predicts a negative outcome (is_churned = 0, customer retained) for instances with similar characteristics.

Positive Contributions: features that positively impact the prediction (ie. the prediction will tend towards 0; customer is likely to retain)

i. **trip_rating**: The higher the trip rating, the more likely a customer is to retain

ii. **age**: Age in this case indicates number of days since the customers first trip, model looked at customers that were active at least once in the 2-month measurement period. High tenure customers will likely have hit product-market fit and likely to retain

iii. **topup_value_per_week:** Customers have the capability to load their wallet and consume of them after rides, this explains that customers who frequently top-up are likely to take rides in the future hence retention

Negative Contributions: Features that negatively impact the prediction (i.e. the prediction will tend towards 1, customer is likely to churn)

i. **price_per_km:** Higher prices per kilometer are associated with a negative outcome.

ii. **bike_cash_count_per_week:** A higher number of bike trips taken on cash (alternative being cashless/credit) counts per week also negatively impacts the prediction.

iii. **days_since_last_activity:** Longer inactivity periods tend to lead to negative predictions.

iv. **perc_failed_requests:** A higher percentage of failed requests negatively impacts the prediction.

v. **active_days_per_week*:** More active days per week negatively impact the prediction.

4.6 Business Impact Evaluation and Economic Theory Interpretation

Table 4.21: Actual customer statuses, promo consumed

Customer Status	Count of Customers	Avg. Promo Value	Total Promo Value
Retained	57,900	8,521	76,342,000
Churned	57,819	8,490	97,787,100
Sum	115,719		174,129,100
	% of Customers		% of Promo Value
Retained	50%		44%
Churned	50%		56%

Table 4.22: Predicted customer status and projected promo

Customer Status	Count of Customers	Avg. Promo Value	Total Promo Value
Retained	51,601	0	0
Churned	64,118	8,522	113,876,500
Sum	115,719		113,876,500
	% of Customers		% of Promo Value
Retained	45%		0%
Churned	55%		100%

Current Promotional Strategy (Actual Results)

Equal Allocation of Promo Across Groups:

i. The current approach equally distributes promotions between retained and churned customers, with each group comprising 50% of the total customer base.
ii. **Retained customers**: 44% of the promo budget, amounting to UGX 76.34 million.
iii. **Churned customers**: 56% of the promo budget, amounting to UGX 97.79 million.

Although churned customers receive a slightly higher share of the promo value, the spend is not targeted, leading to potential inefficiencies.

Retained customers, who are unlikely to churn, consume significant promo resources.

Predicted Promotional Strategy (simulated results based on predictions)

Targeted Allocation Using Churn Predictions:

i. By focusing promo efforts exclusively on customers likely to churn (predicted churn group):
    a. Promo value for retained customers is eliminated (0% of promo spend).
    b. The entire promo budget (UGX 113.88 million) is allocated to predicted churners, who represent 55% of the customer base.
ii. Efficiency Gains:
    a. Total promo spend decreases from UGX 174.13 million to UGX 113.88 million, yielding a savings of 35% (UGX 60.25 million).
    b. The average promo value per churned customer remains consistent (UGX 8,522 vs. UGX 8,490), ensuring retention incentives remain attractive.

Key Implications

i. **Improved Promo ROI**: The targeted strategy significantly reduces wasted promo expenditure on retained customers while focusing resources on at-risk customers who can be influenced to stay.
ii. **Churn Retention Potential**: By reallocating promos, the model ensures that the at-risk group receives adequate incentives to reduce churn likelihood.

iii. **Scalable Cost Efficiency**: The predictive model-driven approach is more cost-efficient, saving a substantial portion of the budget without compromising promotional impact on churn-prone customers.

The observed changes in predicted churn probability can be interpreted through the lens of consumer surplus: as the model shows, features that increase consumer surplus—such as shorter "days since last activity" or higher "bike_count_per_week" correspond to lower churn risk, consistent with the idea that surplus captures the net benefit a user derives from platform engagement. Conversely, variables that erode surplus (for example, higher "price_per_km" or a greater proportion of failed requests) map directly onto increased predicted churn, since diminished net benefit makes alternatives relatively more attractive.

These results can be reframed as empirical estimates of price elasticity within the dataset: the elasticity implied by the coefficient on "price_per_km" indicates how sensitive customer are to fare changes, while the elasticity around "airtime_value_per_week" or "fintech_count_per_week" reflects cross-service complementarities in an all-in-one app context

4.7 Operationalization of the Model

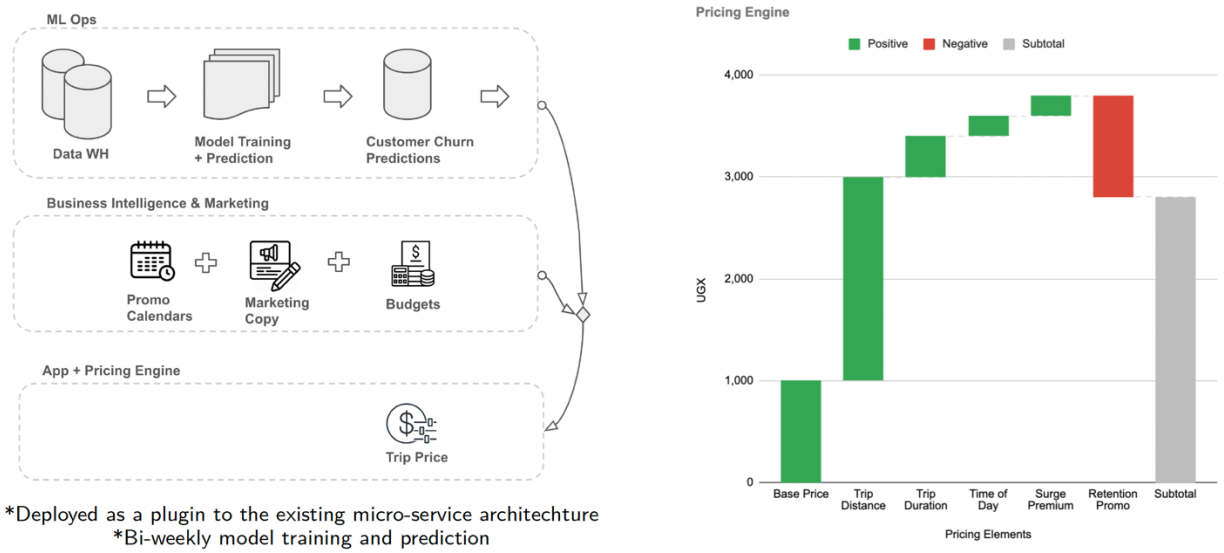

Figure 4.12 Operationalization of the model outputs

Implementing the model in the business operations and especially the pricing strategy shifts from a blanket promotion strategy to a targeted price discrimination approach. By using machine learning to identify customers likely to churn, promotional resources are allocated more effectively, leading to higher ROI. The targeted promos are integrated as a pricing element within the existing ride-hailing pricing model.

Chapter 5: Conclusion, Recommendations, and Future Works

5.1 Conclusions

This study set out to address three critical research questions for a ride-hailing marketplace: (1) What factors affect customer retention? (2) Can machine learning models accurately predict customer churn? and (3) To what extent do dynamic pricing models outperform human agents in maximizing revenues and customer retention?

Through the analysis, several key factors influencing customer retention were identified. The study revealed that days since last activity, payment method preferences (cash versus cashless), active days per week, and trip ratings significantly impact customer churn predictions. Customers with long periods of inactivity, frequent cash payments for bike rides, fewer active days per week, or lower trip ratings are more likely to churn. These insights align with the first research question, highlighting actionable behavioral and transactional patterns that can inform retention strategies.

To address the second research question, a tuned XGBoost classifier was employed to predict customer churn. The model demonstrated strong predictive performance, achieving an accuracy of 73.66%, an F1 Score of 0.7487, and an AUC of 0.81. This shows that machine learning models can effectively predict customer churn with a reasonable degree of accuracy. Moreover, the predictive capability of the model enables businesses to proactively target at-risk customers.

For the third research question, dynamic, data-driven strategies were tested by simulating a discriminatory promotional strategy based on the model's predictions. This approach led to a significant 35% reduction in promotional expenditure (from UGX 174.13 million to UGX 113.88 million) compared to a blanket promotion strategy, while still effectively addressing customer churn. This demonstrates the potential of machine learning to optimize business decisions, outperforming traditional manual strategies in both cost efficiency and impact.

Behavior-aware dynamic pricing research builds on classic revenue management theory by embedding customer behavior and retention considerations into price adjustments (Talluri, Karaesmen, van Ryzin, & Vulcano, 2009). This thesis advances that line of work by integrating

predicted churn propensity, derived from machine-learning models, directly into dynamic pricing rules, thereby aligning price paths with individual retention risk and enhancing total revenue objectives (Talluri, Karaesmen, van Ryzin, & Vulcano, 2009).

Unlike traditional CLV-driven pricing frameworks that treat customers as homogeneous cohorts, the approach in this paper leverages granular customer lifetime value models (Gupta, Lehmann, & Stuart, 2004) to tailor price trajectories for subgroups with distinct future cash-flow profiles. By doing so, we internalize both the expected marginal revenue from a transaction and the marginal cost of retaining a high-risk customer, closing the gap between revenue optimization and retention investment.

Similarly, two-sided marketplace pricing models emphasize balancing cross-side network externalities but often omit dynamic retention costs in their formulations (Rochet & Tirole, 2003). By feeding churn-risk estimates into the platform's objective function, this study enriches game-theoretic pricing frameworks with a behaviour-aware dimension, ensuring that price structures on the rider reflect not only participation incentives but also long-term customer equity.

Finally, while Muth's (1961) rational expectations foundation underpins anticipatory pricing strategies, the empirical results demonstrate how key economic concepts, such as decaying consumer surplus and price elasticity, manifest in real-world, multi-service contexts. This positions the thesis at the intersection of intertemporal optimization, behavioural economics, and marketplace theory, offering a unified framework for future research and practical deployment.

In summary, this research underscores the value of machine learning in understanding and improving customer retention for ride-hailing marketplaces. By identifying key drivers of churn, accurately predicting customer behavior, and implementing dynamic interventions, businesses can maximize both revenue and retention, paving the way for more efficient and sustainable growth strategies.

5.2 Recommendations

Based on the findings of this study, the following recommendations are proposed to improve customer retention and optimize operations for the ride-hailing marketplace:

Develop Targeted Retention Strategies

Focus retention efforts on customers with long periods of inactivity, frequent cash payments, fewer active days per week, or low improve customer experience as measured by trip ratings. Tailored interventions such as personalized promotions or loyalty incentives could address the specific behaviours driving churn.

Promote Cashless Payments

Customers using cash payments exhibit a higher likelihood of churning compared to those using cashless methods. Encouraging cashless transactions through discounts or exclusive benefits could enhance customer retention and streamline operations.

Monitor and Enhance Customer Experience

Since lower trip ratings are associated with a higher risk of churn, continuous monitoring of service quality is essential. Implementing robust feedback mechanisms and addressing complaints promptly can help maintain high customer satisfaction levels.

Leverage Predictive Models in Operational Decision-Making

The predictive accuracy of the machine learning model highlights its potential for operational use. Integrating the churn prediction model into business workflows can enable proactive and cost-effective decision-making, such as targeting at-risk customers for interventions.

The simulation results demonstrate significant cost savings from a discriminatory promotional approach. Transitioning to a data-driven promotion strategy can help optimize promotional spend while maintaining or improving customer retention rates.

Continuous Model Monitoring and Improvement

As customer behavior evolves, the performance of machine learning models may decline over time. Regularly retraining the model with updated data and evaluating its effectiveness will ensure sustained predictive accuracy and relevance.

To effectively integrate machine learning into ride-hailing platforms for dynamic pricing and customer retention, a structured blueprint is essential. This begins with the continuous flow of data from the data warehouse, encompassing customer profiles, historical transaction data, and real-time platform interactions, into the ML Ops pipeline. Within this pipeline, model training and prediction processes are executed, generating customer churn predictions. These predictions then trigger decisions within two key operational arms: Business Intelligence & Marketing and the customer app and the pricing engine. The former utilizes these insights to define promo calendars, marketing copy, and budgets, tailoring campaigns to at-risk segments. Simultaneously, the customer app and the pricing engine directly incorporates these churn prediction as a dynamic element, influencing the final trip price by adjusting promo applications as a pricing lever. To maintain model efficacy and adapt to evolving customer behaviours and market dynamics, a a bi-weekly model refresh cycle should be implemented, ensuring the pricing strategies remain optimized and responsive.

5.3 Future Works

While this study has provided valuable insights into customer retention and churn prediction for a ride-hailing marketplace, several areas remain unexplored and offer opportunities for further research and model enhancement:

Incorporating App Usage Data

Integrating app usage data, such as session frequency, duration, and in-app interactions from platforms like CleverTap, could provide a more comprehensive view of customer activity. These additional behavioural features could improve the model's predictive accuracy by capturing nuances in how customers engage with the app.

Evaluating the Effectiveness of Promotions

A deeper analysis of the effectiveness of promotional interventions in reducing churn is essential. Future work could investigate:

Impact on Retention: Does offering promotions significantly reduce churn, and if so, by how much?

Value Optimization: Are higher-value promotions more effective than lower-value ones? For example, comparing the retention impact of UGX 5,000 versus UGX 10,000 promos.

Promotion Type Effectiveness: Analyzing which types of promotions drive the highest retention, such as:

i. A percentage discount on the next ride.
ii. A cash value promo applicable to future rides.
iii. Bundled promotions offering multiple discounted or free rides.

Understanding these dynamics will help the business optimize its promotional strategy for both cost efficiency and customer impact.

References

Antoniou, A. (2021, October). What is a data model? An anatomy of data analysis in high energy physics. *European Journal for Philosophy of Science, 11*, 101.

Basri, S., Iqbal , H. T., & Naveen, K. (2022). Continuance intentions to use FinTech peer-to-peer payments apps in India. *Heliyon, 8*(11), 116.

Becerril-Castrillejo, I., & Muñoz-Gallego, P. A. (2022). Influence of habitual level of consumption on willingness to pay: A satiation, sensitization, and habituation approach. *International Journal of Hospitality Management, 103*(1), 103-210.

Chen, T., & Guestrin, C. (2016). *XGBoost: A Scalable Tree Boosting System.* New York: Association for Computing Machinery.

Chenavaz , R. Y., & Dimitrov , S. (2025, February). Artificial intelligence and dynamic pricing: a systematic literature review. *Journal of Applied Economics, 28*(1), 46-61.

Cheppala, S., & Lakshya, C. (2024, November). A Comprehensive Approach to Sentiment-Based DynamicPricing:Real-Time Adjustments with Predictive and Ethical Dimensions. *International Journal of Research PublicationandReviews, 5*(11), 6849-6861.

Chernov, A. (2024, February 05). *(GG) MoE vs. MLP on Tabular Data*. Retrieved from arxiv.org: https://arxiv.org/html/2502.03608v1

Copeland, B. J. (2024, December). *Artificial Intelligence*. Retrieved December 2024, from Britannica: https://www.britannica.com/technology/artificial-intelligence

den Boer, A. V. (2015, 5). Dynamic pricing and learning: Historical origins, current research, and new directions. *Surveys in Operations Research and Management Science, 20*(1), 1-18.

DiMicco, J. M., Greenwald, A., & Maes, P. (2003, 7 1). Learning Curve: A Simulation-based Approach to Dynamic Pricing. *Electronic Commerce Research, 3*, 245-276.

Drummond, C. a. (2003, 1). C4.5, Class Imbalance, and Cost Sensitivity: Why Under-Sampling beats OverSampling. *Proceedings of the ICML'03 Workshop on Learning from Imbalanced Datasets*.

Fonti, V., & Belister, E. (2017). Feature Selection using LASSO. *Research in Business Analytics, Vrije Universiteit Amsterdam, 30*(1), 1-25.

Gallego, G., & van Ryzin, G. (1994, 8 8). Optimal Dynamic Pricing of Inventories with Stochastic Demand over Finite Horizons. *Management Science, 40*(8), 1074.

Gao, L., de Haan, E., Iguácel, M.-P., & Sese, F. (2023, March 1). Winning your customers'minds and hearts: Disentangling the effects of lock-in and affective customer experience on retention. *Journal of the Academy of Marketing Science, 51*(2), 334-371.

Gupta, S., Lehmann, D. R., & Stuart, J. A. (2004). Valuing Customers. *Journal of Marketing Research, 41*(1), 7-18.

Gurevich, Y. (2015). Turing's revolution: The impact of his ideas about computation. *Bulletin of the European Association for Theoretical Computer Science, 106*, 1-11.

Haws, K. L., & Bearden, W. O. (2006, October 09). Dynamic Pricing and Consumer Fairness Perceptions. *Journal Of Consumer Research, 33*(3), 304–311.

Jadhav, A., Dhaulakhandi, D., Shandilya, S. K., Malviya, L., & Mewada, A. (2023). Data Transformation: A Preprocessing Stage in Machine Learning Regression Problems. In D. D. Akshay Jadhav, *Artificial Intelligence Techniques in Power Systems Operations and Analysis* (p. 12). Auerbach Publications.

Japkowicz, N. (2000). The Class Imbalance Problem: Significance and Strategies. *Proceedings of the 2000 International Conference on Artificial Intelligence ICAI, https://api.semanticscholar.org/CorpusID:9885187*.

Kephart, J. O., Hanson, J. E., & Greenwald, A. R. (2000, May 30). Dynamic pricing by software agents. *Computer Networks, 32*(6), 731-752.

Kriti. (2019, January 10). *Customer Churn: A Study of Factors Affecting Customer Churn using Machine Learning*. Retrieved from Iowa State University: https://dr.lib.iastate.edu/server/api/core/bitstreams/963c8e0d-4209-4137-9d05-ac20968963f9/content

Lai, P.-L., Hyunmi, J., & Mingjie, F. (2022). Determinants of customer satisfaction with parcel locker services in last-mile logistics. *The Asian Journal of Shipping and Logistics*, 25-30.

Lavazza, L., Morasca, S., & Rotoloni, G. (2023, June). On the Reliability of the Area Under the ROC Curve in Empirical Software Engineering. *Proceedings of the International Conference on Evaluation and Assessment in Software Engineering*, 93–100.

Li, J., Yao, T., & Gao, H. (2010). A Revenue Maximizing Strategy Based on Bayesian Analysis of Demand Dynamics. *Society of Industrial and Applied Mathematics*, 174 - 181.

Malighetti, P., Paleari, S., & Redondi, R. (2007). *Pricing Strategies of low-cost airlines: the Ryanair case.* Brescia, Italy: Department of Economics and Technology Management.

Muth, J. F. (1961, July). Rational Expectations and the Theory of Price Movements. *Econometrica, 29*(3), 315-335.

Nguyen-Phuoc, D. Q., Diep, N. S., & Lester, J. W. (2020). Factors influencing customer's loyalty towards ride-hailing taxi services – A case study of Vietnam. *Transportation Research Part A: Policy and Practice*, 96-112.

Payam, B., Sogand , S., & Cosimo, M. (2025). Enhancing customer retention with machine learning: A comparative analysis of ensemble models for accurate churn prediction. *International Journal of Information Management Data Insights, 5*(1), 100-331.

Rahnama, H., & Pentland, A. (2022, 2 25). *The New Rules of Data Privacy*. Retrieved 3 2022, from Harvard Business Review: https://hbr.org/2022/02/the-new-rules-of-data-privacy

Resnik, D. B. (2018). *The Ethics of Research with Human Subjects - Protecting People, Advancing Science, Promoting Trust.* Springer Cham.

Rochet, J.-C., & Tirole, J. (2003). Platform Competition in Two-Sided Markets. *Journal of the European Economic Association*, 990–1029.

Sasiprapha, A., Khahan, N.-N., & Kaptun, P. (2025). The influence of cost on customer satisfaction in e-commerce logistics: Mediating roles of service quality, technology usage, transportation time, and production condition. *Journal of Open Innovation: Technology, Market, and Complexity, 11*(1), 100-482.

Shin, D. a. (2023). Dynamic Pricing with Online Reviews. *Management Science, 69*(2), 824-845.

Soumi, D. (2022, December). Predicting customer churn: A systematic literature review. *Journal of Discrete Mathematical Sciences and Cryptography, 25*(7), 1965-1985.

Talluri, K., Karaesmen, I., van Ryzin, G., & Vulcano, G. (2009). Revenue management: Models and methods. *Simulation Conference (WSC), Proceedings of the 2009 Winter* (pp. 148-161). Austin: Institute of Electrical and Electronics Engineers.

Appendices

Appendix A: Similarity Report

051234 - Dynamic Pricing Models in Marketplace Environments; The Case of Ride Hailing Business v5.pdf

ORIGINALITY REPORT

14%	12%	12%	8%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www2.mdpi.com Internet Source	3%
2	Submitted to Wright College Student Paper	1%
3	fastercapital.com Internet Source	1%
4	www.coursehero.com Internet Source	1%
5	medium.com Internet Source	<1%
6	www.siam.org Internet Source	<1%
7	"Practical Statistical Learning and Data Science Methods", Springer Science and Business Media LLC, 2025 Publication	<1%
8	www.ncbi.nlm.nih.gov Internet Source	<1%
9	aisberg.unibg.it Internet Source	<1%
10	Submitted to Strathmore University Student Paper	<1%
11	www0.gsb.columbia.edu Internet Source	<1%

12	Submitted to Colorado State University, Global Campus Student Paper	<1%
13	web.media.mit.edu Internet Source	<1%
14	Submitted to Southern New Hampshire University - Continuing Education Student Paper	<1%
15	Submitted to The Scientific & Technological Research Council of Turkey (TUBITAK) Student Paper	<1%
16	su-plus.strathmore.edu Internet Source	<1%
17	Alex Khang, Vugar Abdullayev, Olena Hrybiuk, Arvind K. Shukla. "Computer Vision and AI-Integrated IoT Technologies in the Medical Ecosystem", CRC Press, 2024 Publication	<1%
18	Submitted to York St John University Student Paper	<1%
19	serokell.io Internet Source	<1%
20	H.L. Gururaj, Francesco Flammini, S. Srividhya, M.L. Chayadevi, Sheba Selvam. "Computer Science Engineering", CRC Press, 2024 Publication	<1%
21	link.springer.com Internet Source	<1%
22	H.L. Gururaj, Francesco Flammini, J. Shreyas. "Data Science & Exploration in Artificial Intelligence", CRC Press, 2025 Publication	<1%

Appendix B: Ethical Clearance Release Letter

Strathmore UNIVERSITY

28th August 2024

Evans Munyendo

051234

munyendo.ouma@strathmore.edu

Dear Evans Munyendo,

RE: Dynamic Pricing Models in Marketplace Environments; The Case of Ride Hailing Business

This is to inform you that the Office of Graduate Studies on 12th August 2024 received your request for intervention/assistance following the referral of your matter by the Strathmore University Institutional Scientific and Ethics Review Committee (SU-ISERC) to our Office due to the fact that you stated that you had already collected and/or analysed its data prior to seeking Ethical clearance based on the fact that there was an already existing Non-disclosure & Confidentiality Agreement between you and the company that you were working for (SafeBoda). The agreement's effective date was 24/02/2023. The ethics approval process is ONLY done before any collection of primary or secondary data.

We have taken note of your response and the provided details.

This is a letter for you to proceed with the next steps of your academic requirements.

Please be advised, that in future, all research proposals should be submitted to the SU-ISERC through the RHInnO Ethics platform: https://strathmoreuniversity.rhinno.net/login

Disclaimer: 1) This is not in any way an ethical approval letter. 2) Should there be any legal implications/actions emanating from the research in terms of any ethical violations, you will be personally liable.

Yours sincerely,

Prof. Bernard Shibwabo
Director of Graduate Studies

Ole Sangale Rd, Madaraka Estate. PO Box 59857-00200, Nairobi, Kenya. Tel +254 (0)703 034000
Email admissions@strathmore.edu www.strathmore.edu