VIETNAM NATIONAL UNIVERSITY – HO CHI MINH CITY
INTERNATIONAL UNIVERSITY
DEPARTMENT OF MATHEMATICS

GRADUATION THESIS

DRIVER CHURN PREDICTION FOR RIDE-HAILING SERVICE

Submitted in partial fulfillment of the requirements for the degree of
BACHELOR OF SCIENCE
IN APPLIED MATHEMATICS
SPECIALIZATION IN FINANCIAL ENGINEERING
AND RISK MANAGEMENT

Student's Name: Tran Viet Hang
Student's ID: MAMAIU18079
Thesis Supervisor: Pham Hai Ha, Ph.D

Ho Chi Minh City, Vietnam
February 2023

DRIVER CHURN PREDICTION FOR RIDE-HAILING SERVICE

By

TRAN VIET HANG

Submitted to DEPARTMENT OF MATHEMATICS,
INTERNATIONAL UNIVERSITY, HO CHI MINH CITY
in partial fulfillment of requirements for the degree of
BACHELOR OF SCIENCE
IN APPLIED MATHEMATICS
SPECIALIZATION IN FINANCIAL ENGINEERING
AND RISK MANAGEMENT
February 2023

Signature of Student: _____

TRAN VIET HANG

Certified by: _____

PHAM HAI HA, PhD
Thesis Supervisor

Approved by: _____

Prof. PHAM HUU ANH NGOC, PhD
Head of Department of Mathematics

Declaration

I hereby declare that this thesis represents my own work which has been done after registration for the degree of Bachelor at International University and has not been previously included in a thesis or dissertation submitted to this or any other institution for a degree, diploma, or other qualifications. I have read the University's current research ethics guidelines, and accept responsibility for the conduct of the procedures in accordance with the University's Committee. I have attempted to identify all the risks related to this research that may arise in conducting this research obtained the relevant ethical and/or safety approval (where applicable), and acknowledged my obligations and the rights of the participants.

Acknowledgements

As I come to the end of my university journey, I am filled with a sense of gratitude and appreciation for all the people who have supported and encouraged me along the way. Writing this thesis has been a challenging and rewarding experience, and I could not have done it without the help and guidance of so many individuals.

I would like to start by expressing my deepest gratitude to my thesis advisor, Dr. Pham Hai Ha. Her unwavering support, insightful guidance, and boundless enthusiasm have been an inspiration to me. Her expert knowledge and dedication have made this journey a truly memorable one, and I am deeply grateful for the opportunity to have worked with her.

I would also like to extend my heartfelt thanks to all the lecturers in the Applied Mathematics Department at International University for providing me with the tools and resources needed to pursue my academic interests and achieve my goals.

My family and friends have been a constant source of love, support, and encouragement throughout my university journey, and I cannot thank them enough for their endless belief in me. Their sacrifices and dedication have been the biggest motivation of mine, and I am forever grateful for their love and support.

I would also like to acknowledge and thank my Line Manager, Mr.Dung Dinh. His assistance and support have been instrumental in making this thesis a reality, and I am truly grateful for his contributions.

Finally, I would like to express my gratitude to all the individuals who participated in my research study. Your willingness to share your experiences and knowledge has been invaluable, and I am honored to have had the opportunity to work with you.

DRIVER CHURN PREDICTION MODEL FOR RIDE-HAILING SERVICE

By

TRAN VIET HANG

Submitted to DEPARTMENT OF MATHEMATICS,
INTERNATIONAL UNIVERSITY, HO CHI MINH CITY

* * *

ABSTRACT

In the ride-hailing industry, a high driver churn rate can lead to significant financial losses and negatively impact business growth. To address this issue, it is necessary to develop a driver churn prediction model to help companies take timely and suitable actions to avoid drivers leaving their platforms. The study used a combination of machine learning algorithms, regression model and hybrid model to achieve that objective. Before performing prediction, a technique called multiple time slicing will be introduced, which is a method to help preprocess such a big data set provided by one of 3 biggest ride-hailing companies in Vietnam. This thesis provides valuable insights into driver churn prediction and can serve as a reference for future researches in this field.

Keywords: Driver Churn Prediction, Logistic Regression, Machine Learning, Hybrid Model, Random Forest, Decision Trees, Logit Leaf Model, Ride-hailing.

Contents

List of Figures

Chapter 1

INTRODUCTION

We are now living in a technological era in which various useful apps have been developed to fulfill everyone's needs. Since Uber, a pioneer in the ride-hailing industry, was established in 2009, many other similar companies penetrated to the market with a view to providing fast and convenient online-transportation-booking services to millions of people and significantly changed the online mobility industry globally over the past decade.

Southeast Asia is considered to have a rapid development of the online ride-hailing sector, which is expected to have a market value of 42 billion US dollars by 2025 due to its growing population and high rate of digitalization. Vietnam is one of Southeast Asia's countries that have had the biggest ride-hailing market values in recent years.

Every ride-hailing company has to serve two kinds of customers. One is passengers who have the mobility demand, called riders, the other is people who drive riders from pick-up venues to destination, called drivers. *Churn drivers* are the ones who do not have any completed trips in a week. As a business owner, we desire to attract as many new riders and drivers to join our platform as possible while maintaining our existing customers' base to help the business grow. In other words, we expect to have a relatively low *churn rate* which is defined as the number of churn drivers divided by the number of active drivers during a specific time period. Three

top companies in Vietnamese ride-hailing industry which are Grab, Gojek and Be currently have severe competition in pulling churn drivers to come back to drive with their platforms by designing attractive incentive programs only applied for this special group of drivers. By predicting whether drivers will churn next week, next 2 weeks or even next month or year will help ride-hailing companies take timely and suitable actions to prevent them from leaving their platforms.

There are two research gaps in the area of driver churn prediction: a lack of empirical research in the literature and a lack of consensus over the most effective and accurate churn prediction model. In reality, the bulk of research focusing on forecasting driver turnover in the ride-hailing sector received little attention from academics and practitioners.

Moreover, since some common machine learning models such as Logistic regression or Random Forest only work for static data, it is impossible to apply directly those models on a driver data set that is recorded every week. Hence, preparing usable data which can be used to run machine learning models is of great importance. It is also the most difficult challenge when developing an effective driver churn prediction model.

Most studies on churn prediction focus on a single time window of data, train and evaluate their models on a single time slice, which may not be sufficient to extract meaningful insights from the data. Such an approach cannot capture the evolving conditions and causes of customer turnover over time, which makes it difficult to train a model that can maintain its predictive performance in the future. According to Risselada, Verhoef, and Bijmolt (2010) [1], changing conditions can pose a challenge to training a model that performs well over time.

So as to to fill these research gaps, this thesis aims to (1) *apply multi-time slicing technique to preprocess data so as to build a model to predict whether a driver will churn next week by using different machine learning algorithms* and (2) *use hybrid*

classification algorithm with an aim to improve models' performance using a driver data set from a Vietnamese ride-hailing company.

The remaining sections of the paper are organized as follows. The next part provides a concise literature overview on customer churn modeling, churn prediction in ride-hailing industry and model training with data from multiple periods. Then, both the methodology and data pre-processing procedures are explained, while the results are discussed in the fifth part. Ending my research, the sixth part identifies limitations and suggests future research topics.

Chapter 2

LITERATURE REVIEW

2.1 Customer Churn Prediction Model

In order to compete in an increasingly competitive and global industry, customer churn prediction has gained increased attention during the past ten years (Gordini, 2010). In the management literature, two major philosophies are formed.

The first category consists of conventional categorization techniques such as decision tree (DT) and Logistic regression (LR) (Burez & Van den Poel, 2007 [2], 2009 [3]; Gordini & Veglio, 2013, 2014 [4]; Lemmens & Croux, 2006 [5]; Levin & Zahavi, 2001 [6]; Mozer et al., 2000 [7]; Neslin, Gupta, Kamakura, Lu, & Mason, 2006 [8]; Risselada, Verhoef, & Bijmolt, 2010 [9]; Verbeke et al., 2011 [10]; Verbeke, Dejaeger, Martens, Hur, & Baesens, 2012) [11]. These techniques are quite helpful for analyzing continuous data. They cannot, however, ensure the precision and generalizability of the built models for big scale, nonlinearity, and high-dimensionality.

Instead, the second line of thought is based on the artificial intelligence methods such as neural networks (NNs) (Au, Chan, & Yao, 2003 [12]; Gordini & Veglio, 2013 [13]; Neslin et al. [8], 2006 ; Sharma & Kumar Panigrahi, 2011 [15]; Verbeke et al., 2012) [11], evolutionary leaning (Au et al., 2003 [12]), genetic algorithms (GAs) (Gordini, 2013, 2014 [4, 13]), Random Forests (RF) (Buckinx & Van Den Poel, 2005; Burez & Van den Poel, 2007; Coussement & Van den Poel, 2008; Larivière & Van

den Poel, 2004) [17], improved balanced Random Forests (IBRF) (Xie et al., 2009) [18]. These approaches can overcome the limitations of conventional methods by creating nonlinear mapping capability, resilience, and accurate prediction. In addition, they can result in poor generalization capacity and fuzzy model formation. For instance, the NNs do not directly represent the covered patterns in an accessible way.

However, the previous research revealed that the outcomes of these models are limited and sometimes contradictory, emphasizing the need to build a churn prediction model specifically for ride-hailing companies. Considering the situation when the number of drivers increases gradually over time while the companies' objective is to minimize the incentive amount for drivers, the significance of this need becomes even more apparent.

2.2 Churn Prediction In Ride-Hailing Industry

In terms of predicting customer churn in ride-hailing industry, T.A.H. Tran and K.P. Thai developed a non-contractual churn prediction model for 4-wheel ride-hailing services [21]. The purpose is to predict the churn tendency of the riders within a 3-month threshold. Using the transactional data of Uber and the KDD (Knowledge Discovery in Databases) methodology to conduct the empirical study with the support of python programming language. Three classifiers have been applied including SVM (Support Vector Machine), LR (Logistic Regression), and G-NB (Gaussian Naïve Bayes). The empirical results show that all classifiers achieve higher-than-90% accuracy. Depending upon the characteristics of ride-hailing service, LR is determined as the best model. However, the characteristics of drivers and riders are somehow different. Up to my recent research, there are only a few research papers whose main topic is about drivers.

In terms of driver churn prediction, James Han published his research [22] on his Github website. In his project, churn drivers are defined as the ones who are not active (i.e. took a trip) in the preceding 30 days (from the day the data was ex-

tracted). He preprocessed the data by taking average of the total trips, total distance (in miles), total ratings, etc in the first 30 days. After that, he used XGBoost and Random Forest to build driver churn prediction models. The model using XGBoost can predict the churn by around 78% accuracy with 0.84 AUC and the Random Forest model's accuracy is approximately 76%. Since these two models belong to the tree-based method, their model efficiency is questionable when comparing them to other regression models.

Besides, E.I.Obatoki built a model to predict churn drivers using Logistic regression [23] with data collected from October 2017 to May 2019. He run regression on 4 key categories of driver data, including Hours, Quality, RPH (Rides per Hour) and Earnings with each classified as "Great", "OK", "Needs Attention" depending on the threshold set for each variable. Since his paper's objective is to figure out the key drivers of driver churn, it does not take the time-series factor into account, which makes the nature of this problem become simpler than real-life ones.

2.3 Model Training With Data From Multiple Periods

Few studies have explored the use of multiple time periods in the training data for churn prediction. Gür Ali and Arıtürk (2014) [26] propose a sampling framework that uses multiple samples per customer observed at different times, rather than the conventional approach of one sample per customer at a single moment in time. The authors demonstrate that this approach can improve prediction performance in the banking industry compared to using a single period of training data. However, their study is limited to a single instance with a defined number of periods, and the impact of changing the training set size and number of included periods is unknown.

Two recent studies by Seppala and Thuy (2018) [27] and Leung and Chung (2020) [28] anticipate turnover for housing loan clients and retail bank customers,

respectively, and both show that training on multiple time slices improves prediction performance. However, the multi-slicing technique has not been thoroughly investigated, leaving unknown the impact of training set size and a number of time slices on performance improvement. It is crucial to understand the effects contributing to improved performance and how the amount of time slices used for training affects prediction performance.

Chapter 3

METHODOLOGY

This chapter presents the theoretical background of machine learning models to be used in driver churn prediction, the methodology to transform the data set to be able to apply machine learning models using multiple time-slicing method, the metrics to select significant features to be used in Logistic regression and the evaluation metrics to measure the performance of models.

3.1 Machine Learning Algorithms

3.1.1 Logistic Regression

With the ensembled nature of a classification problem, most classical regression models with continuous dependent variables, including multiple regression and quantile regression, cannot be applied directly to the driver churn prediction problem. A simple and inceptive idea to tackle similar problems, leveraging traditional regression models, is to consider the event

$$A = \{\text{a randomly selected driver is churn}\}$$

and use these models to estimate $\mathbb{P}(A)$, a value not restricted in the set $\{0, 1\}$. For example, the linear probability model (LPM, see (Brooks, 2019) [25]) considers the

OLS regression

$$P_i = \mathbb{P}(y_i = 1|x_i) = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + ... + \beta_k x_{ki} + u_i, \qquad i = \overline{1, N} \tag{3.1}$$

where N is the sample size, $y_i \in \{0, 1\}$ is the value of the binary dependent variable and $x_i = (x_{1i}, ..., x_{ki})$ are values of the k independent variables associated to the i^{th} observation, while $\theta = (\beta_0, \beta_1, ..., \beta_k)$ are the $k + 1$ parameters. In this model, P_i is set equal to y_i since they are not observable. However, this model has several fatal flaws: suppose for example we have estimated the regression

$$P_i = 1.2 - 0.1 x_{1i},$$

where x_{1i} is the employment time in month of driver i, then for drivers employed more than 1 year the churn rate is negative while for drivers employed less than 2 months, the churn rate is greater than 1, which are not suitable since probabilities should lie between 0 and 1. A way to fix this is to set all negative estimated probabilities to 0 and estimated probabilities greater than 1 to 1, but it then results in many observations with an estimated churn rate of exactly 0 and 1. Moreover, it is also not reasonable to predict the churn rate of a given driver to be exactly 0 or 1: in reality, not all drivers employed more than 1 year (less than 2 months) will ever churn (will churn). The Logistic regression overcomes this limitation of the Linear Probability Model (LPM) by using the *Logistic function*

$$F(x) = \frac{e^x}{1 + e^x} = \frac{1}{1 + e^{-x}}, \tag{3.2}$$

with the specified model

$$P_i = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_{1i} + ... + \beta_k x_{ki})}} \Leftrightarrow z_i = \ln \frac{P_i}{1 - P_i} = \beta_0 + \beta_1 x_{1i} + ... + \beta_k x_{ki}, \tag{3.3}$$

here we use z_i as the input of the Logistic function. In words, the Logistic model applies a linear regression with the dependent variable being the log odds of the probability. With this model, the probabilities will never actually fall to exactly

zero or rise to one, although they may come infinitesimally close since 0 and 1 are asymptotes to the Logistic function. Clearly, this model is not linear (and cannot be made linear by a transformation) and thus is not estimable using OLS. Instead, maximum likelihood is usually used. Given that we can have actual zeros and ones only for y_i rather than probabilities, the likelihood function for each observation y_i will be

$$L_i(\theta) = \left(\frac{1}{1+e^{-z_i}}\right)^{y_i} \times \left(\frac{1}{1+e^{z_i}}\right)^{1-y_i} \tag{3.4}$$

The likelihood function that we need will be based on the joint probability for all N observations rather than an individual observation i. Assuming that each observation on y_i is independent, the joint likelihood will be the product of all N marginal likelihoods

$$L(\theta) = \prod_{i=1}^{N} L_i(\theta) = \prod_{i=1}^{N} \left(\frac{1}{1+e^{-z_i}}\right)^{y_i} \times \left(\frac{1}{1+e^{z_i}}\right)^{1-y_i} \tag{3.5}$$

For ease of computation, we will instead maximize the log-likelihood function

$$\begin{aligned}
\ln(L(\theta)) &= \ln\left(\prod_{i=1}^{N} \left(\frac{1}{1+e^{-z_i}}\right)^{y_i} \times \left(\frac{1}{1+e^{z_i}}\right)^{1-y_i}\right) \\
&= \sum_{i=1}^{N} \left[\ln\left(\frac{1}{1+e^{-z_i}}\right)^{y_i} + \ln(\left(\frac{1}{1+e^{z_i}}\right)^{1-y_i}\right] \\
&= \sum_{i=1}^{N} \left[\ln(1+e^{-z_i})^{-y_i} + \ln(1+e^{z_i})^{-(1-y_i)}\right] \\
&= \sum_{i=1}^{N} \left[(-y_i)\ln(1+e^{-z_i}) + (-(1-y_i))\ln(1+e^{z_i})\right] \\
&= -\sum_{i=1}^{N} \left[y_i \ln(1+e^{-z_i}) + (1-y_i)\ln(1+e^{z_i})\right]
\end{aligned}$$

Denote $\ln(L(\theta))$ as LLF, we have to maximize:

$$LLF = -\sum_{i=1}^{N} [y_i \ln(1+e^{-z_i}) + (1-y_i)\ln(1+e^{z_i})] \tag{3.6}$$

To convert this into a loss function (something that must be minimized), we

will simply reverse the sign in Equation 3.6 to have a cross-entropy loss function, denoted as L_{CE}:

$$\begin{aligned} L_{CE} &= \sum_{i=1}^{N} [y_i \ln(1 + e^{-z_i}) + (1 - y_i) \ln(1 + e^{z_i})] \\ &= \sum_{i=1}^{N} \left[y_i \ln(1 + e^{-(\beta_0 + \beta_1 x_{1i} + ... + \beta_k x_{ki})} + (1 - y_i) \ln(1 + e^{\beta_0 + \beta_1 x_{1i} + ... + \beta_k x_{ki}}) \right] \end{aligned} \quad (3.7)$$

The objective of using gradient descent is to identify the ideal set of coefficients that reduces the loss function L_{CE} to its minimum value. Equation 3.7 illustrates the dependence of the loss function L_{CE} on the weights, referred to as $\theta = (\beta_0, \beta_1, \beta_2, ..., \beta_k)$ which contains the coefficients and intercept. Hence, the aim is to figure out a set of weights $\hat{\theta}$ that minimizes the average loss function over all observations:

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^{N} L_{CE}(f(x_i; \theta), y_i) \quad (3.8)$$

This loss function for Logistic regression is conveniently convex. A convex function has at most one minimum point, hence gradient descent beginning at any position is assured to identify the minimum.

The magnitude of the amount to move in gradient descent is the value of the slope $\frac{d}{d\beta} L_{CE}(f(x; \beta), y)$ weighted by a learning rate η. A higher learning rate means that we should move β more on each step. The change we make in our parameter is the learning rate times the gradient. An example of a single variable has

$$\beta^{t+1} = \beta^t - \eta \frac{d}{d\beta} L_{CE}(f(x; \beta), y)$$

We extend the intuition from a function of one scalar variable β to many variables. Then the gradient of a multi-variable function f is a vector in which each component expresses the partial derivative of f with respect to one of the variables. Let $\bigtriangledown$ be

the gradient and F(z) be $f(x;\theta)$, we have:

$$\bigtriangledown L(f(x;\theta),y) = \begin{bmatrix} \frac{\partial}{\partial\beta_1} L(f(x;\theta),y) \\ \frac{\partial}{\partial\beta_2} L(f(x;\theta),y) \\ \vdots \\ \frac{\partial}{\partial\beta_k} L(f(x;\theta),y) \\ \frac{\partial}{\partial\beta_0} L(f(x;\theta),y) \end{bmatrix} \tag{3.9}$$

The final equation for updating θ based on the gradient is thus:

$$\theta^{t+1} = \theta^t - \eta \bigtriangledown L(f(x;\theta),y) \tag{3.10}$$

Logistic regression has been extensively explored and often serves as the standard in churn prediction studies (Coussement et al., 2017) [37]. It is a statistical strategy for classifying samples into two groups using a linear model. As a result of its minimal complexity, its runtimes are short and its interpretability is high. Simultaneously, complex connections are difficult to identify and interactions between characteristics are challenging to account for.

Example 1. *Let's consider a simple example with two features: working years and average weekly rides for a binary classification problem of predicting whether a driver will churn or not. We have a data point as shown below:*

Working years	**Average weekly rides**	**Churn**
3	2	1

The gradient descent algorithm for this simple example is illustrated below:

1. Assume the initial coefficients and intercept are set to 0, and the initial learning rate η is 0.1:

$$\beta_0 = \beta_1 = \beta_2 = 0 \text{ and } \eta = 0.1.$$

2. The single update step requires that we compute the gradient multiplied by the learning rate:

$$\theta^{t+1} = \theta^t - \eta \bigtriangledown_\theta L(f(x_i;\theta),y_i)$$

In this example, there are three parameters so the gradient vector has 3-dimensions, for β_0, β_1, β_2. We can compute the first gradient as follows:

$$\bigtriangledown L_{CE} = \begin{bmatrix} \frac{\partial L_{CE}}{\partial \beta_1} \\ \frac{\partial L_{CE}}{\partial \beta_2} \\ \frac{\partial L_{CE}}{\partial \beta_0} \end{bmatrix} = \begin{bmatrix} (F(z) - y)x_1 \\ (F(z) - y)x_2 \\ F(z) - y \end{bmatrix} = \begin{bmatrix} -0.5x_1 \\ -0.5x_2 \\ -0.5 \end{bmatrix} = \begin{bmatrix} -1.5 \\ -1.0 \\ -0.5 \end{bmatrix}$$

3. Now we compute the new parameter vector θ^1 by moving θ^0 in the opposite direction from the gradient:

$$\theta^1 = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_0 \end{bmatrix} - \eta \begin{bmatrix} -1.5 \\ -1.0 \\ -0.5 \end{bmatrix} = \begin{bmatrix} 0.15 \\ 0.1 \\ 0.05 \end{bmatrix}.$$

4. After one step of gradient descent, the coefficients have updated to be: $\beta_1 = 0.15, \beta_2 = 0.1$ and $\beta_0 = 0.05$.

5. We repeat the process of calculating the predicted probabilities, calculating the gradient, and updating the parameters until the difference between the current and previous parameters is smaller than a threshold or the maximum number of iterations is reached.

3.1.2 Random Forest

Random Forests algorithm is a method that combines multiple decision trees to produce a prediction by taking the most frequently occurring class. To prevent over-fitting, the method uses bagging (bootstrap aggregation) to average the opinions of many trees constructed from bootstrap samples. The prediction for a new data point is typically made by averaging the predictions of the individual trees in the forest. Various studies have demonstrated the strong and reliable predictive performance of Random Forests in churn prediction (Buckinx & Van den Poel, 2005; Burez & Van den Poel, 2009; Coussemant & Van den Poel, 2008; 2009; Verbeke et al., 2012 [11]).

The fundamental concept behind the Random Forest method is as follows (Qiannan Zhu et al 2019)[20]: Firstly, feature selection is performed on the decision tree to increase the purity of the categorized data set. Here, the GINI index is adopted as the criterion for measuring purity:

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk}), \tag{3.11}$$

where $\hat{p}_{mk}$ represents the percentage of training observations in the m^{th} region belonging to k^{th} class.

The formula can be interpreted as:

$$G = \sum_{k=1}^{K}(\hat{p}_{mk} - \hat{p}_{mk}^2) = \sum_{k=1}^{K} \hat{p}_{mk} - \sum_{k=1}^{K} \hat{p}_{mk}^2 = 1 - \sum_{k=1}^{K} \hat{p}_{mk}^2 \tag{3.12}$$

If we have a training data set with N observations and M features, the Random Forest algorithm follows the steps below:

1. **Random Record Selection**: A subset of n observations is randomly chosen with replacement from the entire training data set to create the training data for a single decision tree. Typically, each tree is trained on about 2/3 of the total training data.

2. **Random Variable Selection**: k features are selected from all the available predictor variables M, and the best split is made on these k features to split a node. By default, the value of k is the square root of the total number of features in the dataset.

3. Among the values of each of the k features selected, the best binary split is chosen based on the Gini impurity, and features with the best index values are chosen.

4. The tree is grown to its maximum size according to the stopping criterion chosen.

Typically, node splitting is terminated when one of the following conditions is met:

- The number of samples in the node to be divided is less than a predetermined threshold.
- All samples within the node are of the same class.

5. Let the tree remain unpruned. In other words, it is advised not to cut trees while they are growing in Random Forests.

Following the building of Random Forests, a sample is labeled according to the Majority Voting rule, i.e., it is labeled with the ensemble trees' most popular class. In [40], it is demonstrated that Random Forest does not overfit when more trees are added; rather, its generalization error moves toward a limiting value.

The Random Forest model is an integrated model based on the cart decision tree, which enhances the accuracy of predictions while ignoring the multicollinearity issue between variables. However, with high-dimensional data, the model is complicated and computationally intensive, and it lacks generalization capability.

Example 2. *Let's consider a simple example with two features: average weekly requests and average daily online hours for a binary classification problem of predicting whether a driver will churn or not. We have 5 observations as shown below:*

Average weekly requests	**Average daily online hours**	**Not Churn**
3	3	1
4	4	0
5	3	0
6	2	1
8	2	1

In this example, the features are represented by a 2-dimensional array X and the target by a 1-dimensional array y. The target shows whether the driver is a churner.

The data consists of five observations, each with two attributes (1 for not churn, 0 for churn). Apparently, 4 cutpoints must be examined to determine the best one. The difference between classification and regression problems begins at this point. For the first cutpoint, it will divide the sample set into two subsets as Figure 3.1.

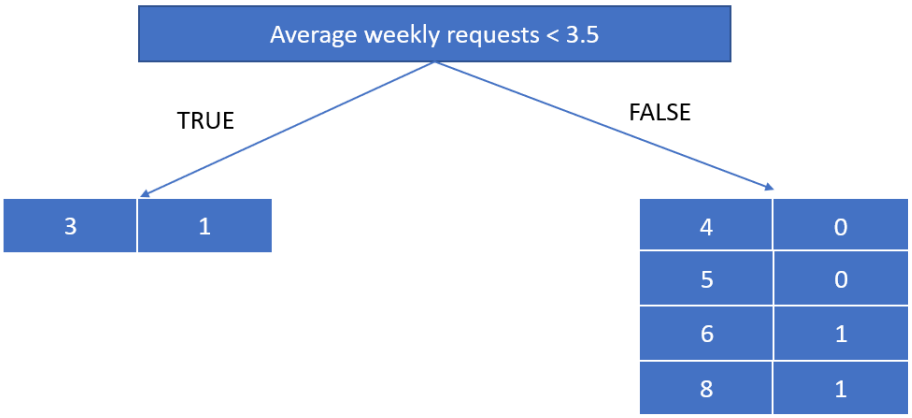

Figure 3.1: First cutpoint of *Average weekly requests*

Recall that with Gini index, the proportion of observations within k^{th} class needs to be calculated. Therefore, the number of observations in each class is counted for each region (the two regions are $\{X \mid \textit{Average weekly requests} \geq 3.5\}$ and $\{X \mid \textit{Average weekly requests} < 3.5\}$).

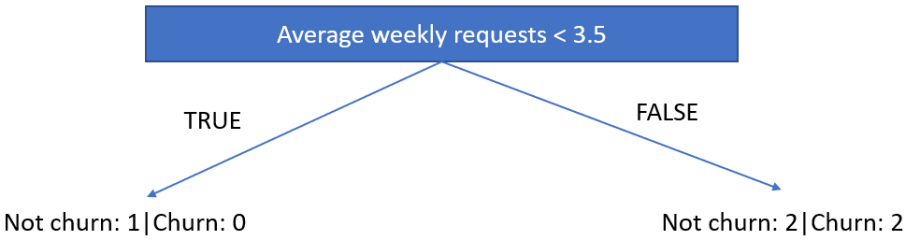

Figure 3.2: Classification of observations in each region

Recall that the formula for Gini index is simplified as: $G = 1 - \sum_{k=1}^{K} \hat{p}_{mk}^2$. Applying the formula to this situation, there are only two types of response variables, which are *not churn* and *churn*. Hence, K = 2. For the region $\{X \mid$

Average weekly requests $< 3.5\}$, Gini index equals:

$$G_{\{X|Average\ weekly\ requests<3.5\}} = 1 - \hat{p}^2_{notchurn} - \hat{p}^2_{churn} = 1 - \left(\frac{1}{1+0}\right)^2 - \left(\frac{0}{1+0}\right)^2 = 0$$

Likewise, Gini index for the region $\{X \mid$ *Average weekly requests* $\geq 3.55\}$ equals:

$$G_{\{X|Average\ weekly\ requests\geq 3.5\}} = 1 - \hat{p}^2_{notchurn} - \hat{p}^2_{churn} = 1 - \left(\frac{2}{2+2}\right)^2 - \left(\frac{2}{2+2}\right)^2 = 0.5$$

Nevertheless, those are not Gini index of the cutpoint $\{X \mid$ *Average weekly requests* $<$ $3.5\}$. Since the two regions contain different numbers of observations, it is appropriate to use the weighted average of Gini index. The weight of each region is simply its number of observations divided by the total number of observations in the whole set. Specifically, weight of $\{X \mid$ *Average weekly requests* $< 3.5\}$ is $\frac{1}{5}$ and weight of $\{X \mid$ *Average weekly requests* $\geq 3.5\}$ is $\frac{4}{5}$. Therefore, the weighted average Gini index of the cutpoint is:

$$G = 0 \times \frac{1}{5} + 0.5 \times \frac{4}{5} = 0.4$$

The procedure remains the same for all remaining cutpoints. Table 3.1 summarizes Gini indexes of all cutpoints.

Cutpoints	**Average weekly requests**
1	0.4
2	0.4667
3	0.2667
4	0.4

Table 3.1: Gini indexes of all cutpoints

As shown in the table 3.1, *Average weekly requests* with cutpoint = 3 has the lowest Gini (0.2667), hence, (*Average weekly requests* ≥ 5.5 is the best cutpoint of the first iteration. The first image of the tree is illustrated as Figure 3.4.

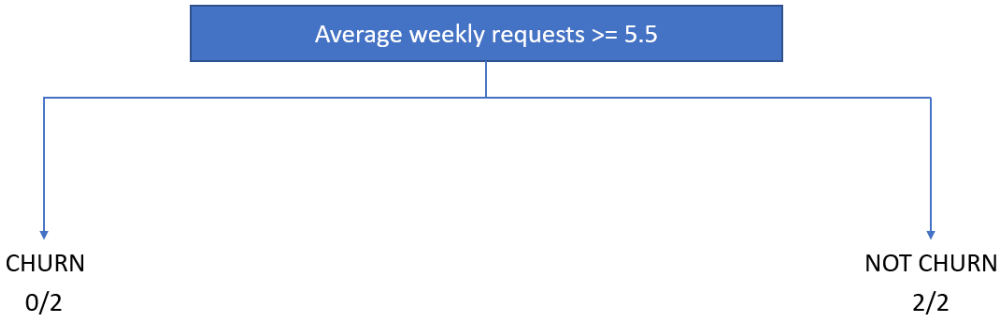

Figure 3.3: A tree created after the first iteration

Stopping criteria: The Gini index of the considered set must be computed and compared with the Gini index of the optimal split. The next iteration will be implemented to show how stopping criteria work. The dataset is now split into two subsets: $\{X \mid \textit{Average weekly requests} \geq 5.5\}$ and $\{X \mid \textit{Average weekly requests} < 5.5\}$. $\{X \mid \textit{Average weekly requests} < 3.5\}$ is firstly considered.

Average weekly requests	**Average daily online hour**	**Not churn**
6	2	1
8	2	1

Table 3.2: Observations with *Average weekly requests* ≥ 5.5

Following the same algorithm as the first iteration, Gini index of the whole subset $\{X \mid \textit{Average weekly requests} \geq 5.5\}$ is computed. The subset contains 2 observations with responses 1 (not churn) and no response with 0 (churn). Hence, its Gini index is:

$$1 - \left(\frac{2}{2+0}\right)^2 - \left(\frac{0}{2+0}\right)^2 = 0$$

Obviously, $0 < 0.2667$, for that reason, splitting is continued with cutpoint ($\textit{Average weekly requests} \geq 7$).

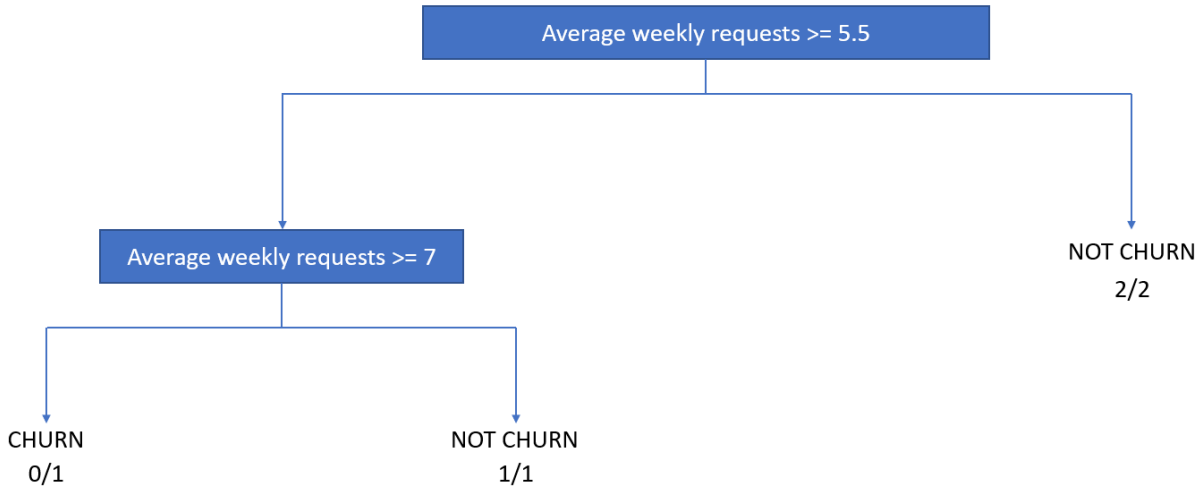

Figure 3.4: Illustration of the tree after two iterations

3.1.3 Logit Leaf Model

Logistic regression and Random Forest are two of the most common models for predicting churn drivers due to their high predictive performance and high level of comprehensibility. Despite these advantages, Random Forest tends to struggle with linear relationships between variables, whereas Logistic regression has difficulty with interaction effects. To improve data classification, the **logit leaf model (LLM)** is presented as a new hybrid technique. The concept underlying the LLM is that alternative models created on subsets of data rather than the complete dataset result in better predicting performance while preserving the interpretability of models constructed in the leaves. The LLM is comprised of two phases: (1) segmenting drivers into different groups and (2) applying Logistic regression and Random Forest to each driver group. This method is called **hybrid classification approach**, introduced by Arno De Caigny, Kristof coussementt, Koen W. De Bock in 2018[29].

In this thesis, K-Means algorithm was used to segment drivers in the first phase. Suppose we are given a set of points that we want to classify into k distinct groups, called *clusters*. Assume that k is known *a priori* and the distance between any two

points can be calculated, the K-Means algorithm proceeds as follows:

- Step 1: Choose k points and set them as *centers* of the clusters.
- Step 2: Assign each remaining point p to the cluster with center closest to p.
- Step 3: Recalculate the center for each cluster. Usually the updated center is chosen to be the *centroid* of the cluster.
- Step 4: If the distance between any center before and after updating is significant (e.g. greater than a predetermined threshold), go back to Step 2. Otherwise, stop the algorithm.

After segmenting drivers into some clusters, we apply the Logistic regression and Random Forest predictive model for each cluster to compare models' performance.

3.2 Multiple Time Slicing

To use multiple time slices for training, time-stamped transactional data needs to be managed appropriately. The author introduces a notation for time slices and explains the idea of out-of-period testing in Section 3.2.1, before discussing the multi-slicing technique in Section 3.2.2.

3.2.1 Time Slices

A time slice is a subset of the accessible data for a certain time period. In between the feature and label windows is the predicting origin t, which serves as a point of reference.

A time window w of earlier data is employed relative to time t to calculate characteristics of consumer behavior during this period. For label construction addressing driver turnover, a time frame v of the following data is utilized. A time slice's driver base is the set *C(t)* of active drivers at the origin of forecasting at time t. Features and accompanying labels are calculated for these drivers.

Per driver at time t, the feature matrix X_t comprises of non-time-varying features and time-varying features. Typically, non-variable characteristics consist of client master data such as city, vehicle brand, vehicle model, device type. Number of online hours, trips, trips' fare, etc that occurred within the time window w leading up to the forecasting origin t create time-varying characteristics. For each driver $i \in C(t)$ examined at time t, a label y_{it} is generated using the transactional data collected during the subsequent time frame v. The label y_{it} is 1 if driver i's behavior during v implies *not churn*, and 0 otherwise.

The sample size is dependent on the number of active drivers $\mid C(t) \mid$ over time. Various observations of the same driver may appear in distinct time slices.

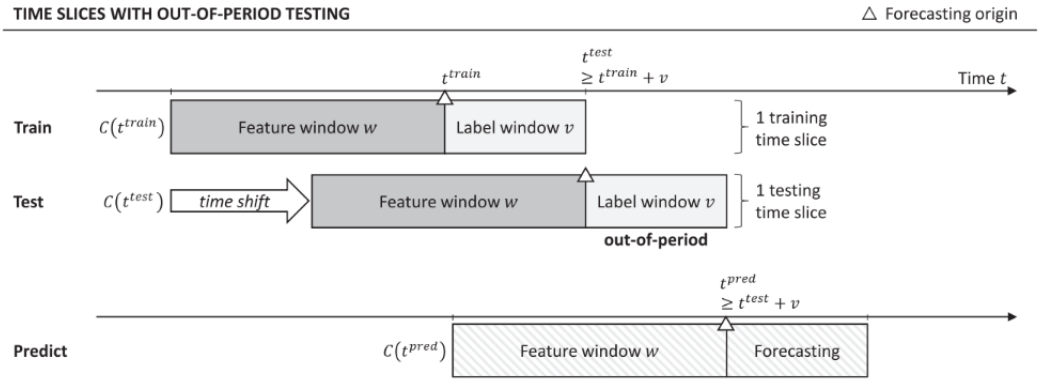

Figure 3.5: Time slice concept illustrated with out-of-period testing

In Figure 3.5, I illustrate the time slice idea by depicting the training, testing, and prediction time slices. The forecasting sources t^{train}, t^{test}, and t^{pred} are distinguished.

One time slice is utilized to train the model with window w-derived features and window v-derived labels. A more recent time slice is utilized for testing, ensuring that label windows do not overlap. The testing time slice is moved by v so that the test set's forecasting origin is $t^{test} \geq t^{train} + v$. This causes testing to occur outside of the label period window. The image also demonstrates that projecting a future that has not yet occurred may begin as early as $t^{pred} \geq t^{test} + v$, after known data has been utilized for training and testing.

3.2.2 Multi-slicing

The concept of multi-slicing becomes feasible when transactional data is divided into time periods. This concept enables the compilation and merging of multiple time slices to form a single training dataset.

Drivers are observed at different periods using multi-slicing t_k^{train}. The characteristics and labels of each time slice $k \in \{1, ..., K\}$ represent a snapshot of the drivers at the beginning of forecasting t_k^{train}. The most recent slice displays drivers at t_K^{train}, whereas the oldest slice displays customers at t_1^{train} displaced by $K-1$ periods. A stacked feature matrix $X_{K,t^{train}} = [X_{t_1^{train}}, ..., X_{t_K^{train}}]^T$ and a stacked label vector constitute the training data set. $X_{K,t^{train}}$ contains up to k rows of $\Sigma_k \mid C(t_k^{train}) \mid$. Note that $K-1$ extra data periods are utilized for training on K time slices rather than just one.

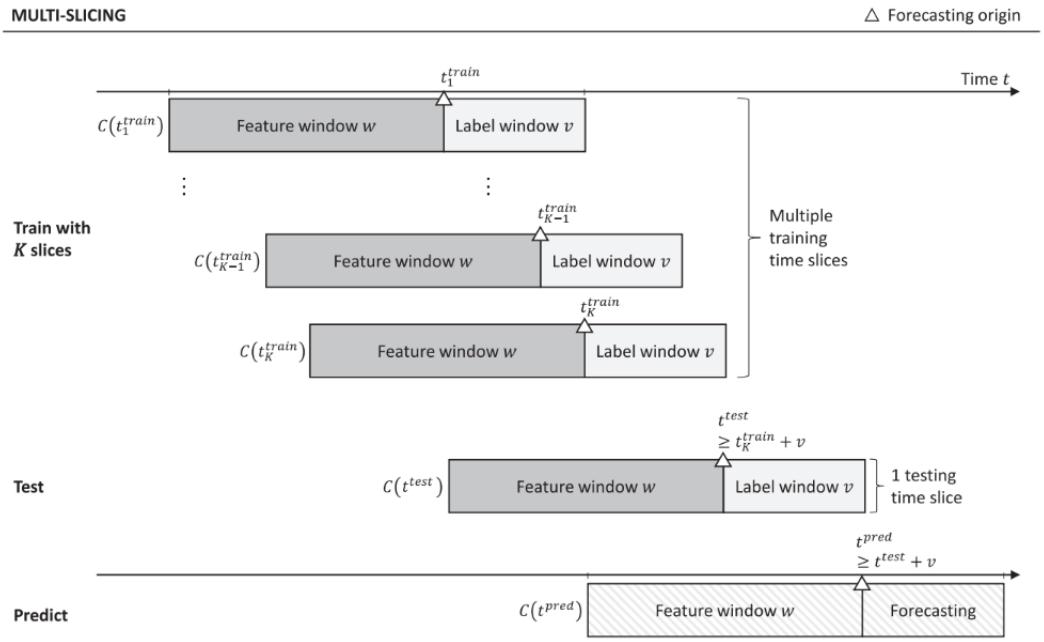

Figure 3.6: Training on multiple time slices

The training set for multi-slicing is depicted in Figure 3.6 as consisting of numerous time slices. Included also are the time slices necessary for testing and probable prediction.

The approach of multi-slicing is distinct from under-sampling or over-sampling techniques. It allows the use of more authentic observations from historical data for both minority and majority classes without changing the essential class distribution. By dividing the data into time slices, more samples can be generated to deal with the issue of extremely scarce or missing data, while only slightly extending the time range used. Hence, I follow the recommendation of Weiss (2004)[30] to use all available data and avoid discarding any. Multi-slicing enables learning from both past churners and non-churners, as well as the most recent individuals.

3.3 Feature Selection By Weight Of Evidence And Information Value

Weight of Evidence (WOE) and **Information Value** (IV) are commonly used for feature selection before applying Logistic Regression, Random Forest [38], or other machine learning models.

3.3.1 Weight Of Evidence

At the data pre-processing stage to handle classification issues, the transformation of features is of primary necessity. Various classification methods favor continuous attributes over discrete attributes, and the distance between data points cannot always be determined if attribute values are not continuous and standardized. The Weight Of Evidence (WOE)[31] possesses a number of desired properties that make it a highly effective tool for the transformation of attributes. It is usually used to convert nominal variables to continuous variables in labeled data sets, i.e. in supervised learning.

The Weight of Evidence (WOE) metric is used to choose the predictor variables having the highest predictive power for the dependent variable (in this study, not churn next week event). WOE illustrates, in general, how effectively an independent

variable predicts a dependent variable, or how much evidence the independent variable has for fluctuations in the dependent variable. In situations involving binary classification, WOE might be described as:

$$WOE_i^A = \ln\left(\frac{N_i^A/SN}{P_i^A/SP}\right), \tag{3.13}$$

where

$$SN = \sum_{i=1}^{n} N_i^A \text{ and } SP = \sum_{i=1}^{n} P_i^A \tag{3.14}$$

Equation (3.13) provides a definition of the weight of evidence (WOE) for a specific attribute A and its i^{th} value, where N_i^A and P_i^A are the counts of negatively and positively labeled data points, respectively. The total number of negatively labeled data points is denoted by SN, while the total number of positively labeled data points is represented by PN. Moreover, there are n values for the attribute A.

In Equation (3.13), N_i^A and P_i^A should have values greater than zero, as they represent counting numbers. Thus, the constraints become $N_i^A > 0$ and $P_i^A > 0$. The second part of Equation (3.13) shows that the WOE has two parts: a variable part that corresponds to the data points with a specific attribute value, and a constant part that applies to the entire training dataset. These values are calculated during data preparation and do not depend on the classification method used.

The following example shows WOE calculation:

Example 3. *Suppose there is a training data set for drivers in the ride-hailing industry that consists of 50,000 data points, with 10,000 labeled as "churn" and 40,000 labeled as "not churn". One of the attributes in the data set is the "Number of completed trips per week" (nbtrips), which takes on 70 different values. Our goal is to calculate the WOE for this attribute when its value is 20 (meaning that 1,000 data points have the value of 20), given that 700 of these data points are labeled negatively and 300 are labeled positively. Table 3.3 shows the parameters needed to*

compute the WOE, which can be done using equations (3.13) and (3.14). Equation (3.15) provides an example of how to calculate the WOE.

Parameter	**Description**
SP=10,000	Constant for the whole data set
SN=40,000	Constant for the whole data set
n=70	Constant for the whole data set
$P_i^{nbtrips} = 300$	Applies only for the value 20 (trips) of the attribute *nbtrips*
$N_i^{nbtrips} = 700$	Applies only for the value 20 (trips) of the attribute *nbtrips*

Table 3.3: Values of the parameters for the calculation of WOE in Example 3

$$WOE_i^{nbtrips} = ln(700/300) - ln(40000/10000) = 0.8473 - 1.3863 = -0.539 \quad (3.15)$$

The resultant number indicates that the group to which the WOE is applied has a greater risk than the average or a negative weight of evidence. The WOE for any group with average chances is zero, as the constant and variable components of (3.13) would be roughly equivalent.

To calculate WOE for a continuous variable, we will first *discretize/categorize* it by dividing its range into n intervals of equal size, and then apply the formula (3.13) on the transformed variable. This procedure, known as *fine classing*, is being widely used in the context of credit risk modelling, see ([32]) for a detailed treatment on the topic.

The WOE has a linear connection with the Logistic function, making it an ideal instrument for attribute transformation when Logistic regression is employed.

There are some primary reasons why WOE is an effective metric:

- It delivers a simple and clear estimation of the relative risk associated with the different attribute values.
- Indicates the riskier and safer value groupings.
- It may be utilized as a very useful tool for the straightforward transition of

qualities from one kind to another. This is especially helpful for converting multi-valued non-numerical (i.e. nominal) qualities into numerical attributes with continuous values (the WOE values).

- After transforming the characteristics, the groupings of values with comparable relative risk were readily identifiable. This feature may be used to arrange many values into fewer bins. Using this property, the number of distinct groups for the characteristic "Number of completed trips per week" in Example 3 might be greatly decreased. One group would correspond to numerous values with equal relative risk, and the group would be represented by the average WOE of those values.

- Using the WOE of an attribute's values, the information value (i.e. predictive power) of an attribute might be evaluated.

A few disadvantages of WOE exist, in contrast to its many positive advantages. Before calculating the WOE transformation to an attribute, it is necessary to handle the following:

- WOE considers just the relative risk, not the proportion of data points with a certain attribute value. In Example 3, $P_i^{nbtrips} = 300$, $N_i^{nbtrips} = 700$ and $WOE_i^{nbtrips}$ = -0.539. We could obtain the same value for WOE if we change $P_i^{nbtrips}$ to 3 and $N_i^{nbtrips}$ to 7 although these examples are very different in terms of the proportion of data points from the whole data set. This issue must be handled using additional statistical approaches, such as *information value*, which will be discussed in the next section.

- The weight of evidence (WOE) only evaluates the discriminability of a single attribute and does not take into account the discriminability of a combination of attributes. For instance, an attribute X may have a low WOE for its value A, but when combined with a second attribute Y that has a value B, the combination $X{=}A$ and $Y{=}B$ may have a high WOE and be very useful for classification. Therefore, assigning a low WOE to $X{=}A$ should be done carefully. It is better to first identify the interacting qualities using a

suitable method and then incorporate the known interactions into the model. Non-parametric methods like classification trees and neural networks are well-suited for identifying and handling interactions. Classification trees can be used to discover the interactions and then another classification technique can be used to build a prediction model. Alternatively, one can segment the data into groups and apply a separate classification model to each group or create interaction variables and apply a single model to the entire data set to account for interactions (L.C. Thomas, D.B. Edelman, J.N. Crook, 2002) [36].

3.3.2 Information Value

The *information value* metric might be used to quantify the predictive power of a certain property (R. Anderson, 2007) [33]. It is a valuable metric since it may be computed during the pre-processing phase and utilized for feature selection by removing features with extremely low information value. The computation may be performed using equation (3.13). Here, F^A is the information value of attribute A, while the definitions of the other parameters remain unchanged from (3.13) and (3.14).

$$F^A = \sum_{i=1}^{n} \left[\left(\frac{N_i^A}{SN} - \frac{P_i^A}{SP} \right) \times WOE_i^A \right] \tag{3.16}$$

F^A values are always positive and may exceed 3 for highly predictive characteristics. In general, variables with an information value of less than 0.1 are regarded weak predictive, whereas those with an information value of higher than 0.3 are desirable and likely to be incorporated into scoring models. Note, however, that weak qualities may be useful when combined with other traits, or that their individual values may give predictive power as dummy variables. (Eftim Zdravevski, Petre Lameski, Andrea Kulakov, 2011) [31].

The first disadvantage of WOE indicated at the conclusion of the preceding section is addressed with the information value in the following manner. In (3.13), the first term of the product comprising N_i^A and P_i^A tends to zero when they are very small, indicating that the entire product for a certain value of i tends to zero re-

gardless of the value of WOE_i^A. In actuality, this indicates that this single instance will not have a major impact on the whole amount in (3.13), i.e. the information value of attribute A.

For continuous variables, IV is calculated following the *fine classing* procedure mentioned in the previous section. To ensure that fine classing has minor effects on IV calculation, the common practice is to determine multiple IVs associated with different bin division counts and choose the highest IV as the final IV for consideration of the variable.

Note that the IV of a categorical variable may equal infinity if there exists any value (bins for continuous variables) whose associated observations share the same label. Usually, the observation counts for these cases are insignificant and therefore, they can be omitted from the IV calculation. The resulting IV, may we call it the *adjusted IV* of the variable, can then be used for variable selection.

The *information value* is sensitive to how the attribute is classified and the number of groups, but regardless of how the data are organized, it will provide the same result. However, it might be challenging to understand due to the absence of accompanying statistical testing. In general, it is recommended to utilize the information value and/or chi-square test to evaluate certain characteristics.

3.4 Evaluation Metrics

Precision, Recall, F1 score, area under the receiver operating curve (AUC), Gini and Kolmogorov-Smirnov that were established in the churn prediction domain (Coussementt et al., 2017; De Bock & Van Den Poel, 2012; De Caigny et al., 2018; Verbeke et al., 2012 [11]) are chosen to evaluate my model. Note that most of them are derived from the confusion matrix. It quantifies, for classification tasks with binary labels (1: positives = not churn, 0: negatives = churn), the number of correct predictions (true positives (TP) and true negatives (TN)) and wrong pre-

dictions (false positives (FP) and false negatives (FN)). The sample distribution in the confusion matrix relies on the probability threshold of the classifier's decision function. The following threshold-dependent metrics can be calculated as follows:

$$TruePositiveRate(TPR) = \frac{TP}{TP + FN}, \tag{3.17}$$

which is also known as *Recall*, measuring the ability of the model to find all of the positive instances.

$$FalsePositiveRate(FPR) = \frac{FP}{FP + TN}, \tag{3.18}$$

Precision is defined as the number of true positive predictions (i.e., the number of instances correctly predicted as positive) divided by the total number of positive predictions made by the model. The formula for precision is given by:

$$Precision = \frac{TP}{TP + FP} \tag{3.19}$$

The F1 score balances the trade-off between precision and recall and is defined as the harmonic mean of precision and recall. The formula for the F1 score is given by:

$$F1 - Score = \frac{2(Precision \times Recall)}{Precision + Recall} \tag{3.20}$$

Note that:

- A model will obtain a **high F1 score** if both Precision and Recall are high.
- A model will obtain a low **low F1 score** if both Precision and Recall are low.
- A model will obtain a **medium F1 score** if one of Precision and Recall is low and the other is high.

The receiver operating curve (ROC) takes into account the trade-off between these two metrics, with an increased TPR resulting in a greater FPR. The area under the curve gives a performance summary across thresholds that may be used to compare

models (Egan, 1975) [35] and is derived by approximating:

$$AUC = \int_0^1 TPR\, dFPR \tag{3.21}$$

The resultant AUC value demonstrates the model's ability to differentiate between churners and non-churners. It may be understood as the likelihood that a randomly selected non-churner would have a better score than a randomly selected churner. A baseline model that randomly allocates the class has an AUC score of 0.5.

Gini [34] is also commonly used to evaluate the performance of a binary classification model. It is a measure of inequality in a distribution. The Gini index for a binary classification problem can be calculated as follows:

$$Gini = 2 \times AUC - 1, \tag{3.22}$$

where AUC, or the Area Under the Receiver Operating Characteristic (ROC) Curve, is a common performance measure used for binary classification models. It measures the ability of a model to distinguish between positive and negative classes and is calculated as the area under the ROC curve, which plots the true positive rate (TPR) against the false positive rate (FPR) as the threshold for classifying a positive example is varied.

The Gini index and AUC are related, as the Gini index can be calculated from the AUC by the formula given above. However, they have some important differences. The Gini index is a single value that summarizes the model performance, while the AUC is a curve that provides a more nuanced view of performance over a range of thresholds.

The *Kolmogorov-Smirnov (KS)* index is a statistical test that measures the maximum difference between the cumulative distribution functions (CDFs) of two samples. In the context of binary classification, the KS index is used to compare the distributions of predicted probabilities of the positive class for samples of the positive

and negative classes. The KS index is calculated as follows:

$$KS = max \mid (F_{1x} - F_{2x}) \mid, \tag{3.23}$$

where F_{1x} and F_{2x} are the CDFs of the positive class predicted probabilities for the positive and negative classes, respectively. The maximum difference between the CDFs, which is represented by the KS index, is a measure of the separation between the distributions of predicted probabilities for the positive and negative classes.

A high KS index indicates that the model is able to distinguish well between the positive and negative classes, while a low KS index indicates that the model is not able to effectively distinguish between the classes. The KS index is widely used in financial modeling and credit scoring to evaluate the performance of binary classification models.

It is important to note that the Kolmogorov-Smirnov test is only one of many methods for evaluating the performance of binary classification models, and it should be used in conjunction with other metrics such as accuracy, precision, recall, and the receiver operating characteristic (ROC) curve to get a comprehensive view of model performance.

Chapter 4

EMPIRICAL STUDY

This chapter describes in detail the driver data set and its attributes. Some preprocessing steps will also be introduced to generate the dependent variable as well as independent variables. The exploratory data analysis section will be included in this chapter to give you a better understanding of the distribution of the data before ending with splitting the preprocessed data into train set and test set.

4.1 Data Description

The driver data set contains 761,103 rows and 33 columns. The data collected spans 21 weeks (from August 1 to December 19, 2022). Each row records information of **one driver** in **one observed week**, including his/her personal information and performance during that week. Personal information of one driver such as city, vehicle model, activated date, etc does not change over time since it is kept in the data mart after he/she registers a driver account at the ride-hailing company. However, the performance of a driver per week such as the number of completed trip, total trip amount, total traveled distance, etc changes every week. some of the data attributes, description, number of distinct values, missing values and examples are shown in Figure 4.1. Full data attributes can be found in the Appendix.

Variable's name	Data type	Description	Num_distinct_values	Num_missing_values	Example
week	datetime64	Observed week	21	0	01/08/2022
driver_id	int64	Driver ID	62,237	0	342
activated_date	datetime64	Date when driver's account is verified	1,114	0	20/07/2019
ops_city_id	int64	City where driver lives (189 = HCMC, 190 = Hanoi)	2	0	189
vehicle_model	string	Driver's vehicle model	377	65	Avanza
sum_sh	float64	Total online hours in the observed week of a driver	238,133	0	4.46
segment	string	Driver segment (High full-time, Full-time, Medium, Low medium, Part-time) based on total completed trips in the observed week	5	0	Part-time
completed_trip	float64	Number of completed trips in the observed week of a driver	292	29,057	7
total_fare	float64	Total fare of completed trips of a driver in the observed week (VND)	66,993	29,057	845,000
gmv_aftertax	float64	Total fare after tax of completed trips of a driver in the observed week (VND)	596,034	29,057	797,745
total_ETA	float64	Total estimated time arrival of a driver in the observed week (minute)	1,966	50,895	52
total_ATA	float64	Total actual time arrival of a driver in the observed week (minute)	1,884	76,363	80
trip_time_hour	float64	Total time a driver drives from the pick-up venue to the destination in the observed week (hour)	3,862	75,883	3
en_route_hour	float64	Total time a driver drivers from his/her location to the pick-up venue in the observed week (hour)	1,885	75,875	1
distance_travelled	float64	Total travelled distance of a driver in the observed week (km)	88,978	29,057	90
driver_trip_amount	float64	Total amount that a driver actually earns in the observed week (VND)	243,133	29,057	525,163
weekly_amount	float64	Total weekly incentive amount that a driver earns in the observed week (VND)	12,738	476,958	110,000
loyalty_amount	float64	Total incentive amount that a driver earns from the loyalty program in the observed week (VND)	51,174	74,786	402,345
not_churn	int64	Churn status of the observed week of a driver (1 = not churn, 0 = churn)	2	0	1

Figure 4.1: Random data attributes, description, number of distinct values, missing values and example of driver data set

4.2 Data Pre-processing

Since the churn status of a driver only depends on the number of completed trips in a week, in more general, the performance of a driver in a specific week, static variables recording drivers' personal information (*ops_city_id*, *activated_date*, *device_type*, *ftt_time*, *vehicle_brand*, *vehicle_model*) are dropped. Note that in this ride-hailing company, *segment* of a driver in the observed week is defined as follows:

- High full-time: Total weekly online hours ≥ 60 ($sum\_sh \geq 60$)
- Full-time: Total weekly online hours ≥ 40 ($sum\_sh \geq 40$)
- Medium: Total weekly online hours ≥ 20 ($sum\_sh \geq 20$)
- Low medium: Total weekly online hours ≥ 10 ($sum\_sh \geq 10$)
- Part-time: Total weekly online hours < 10 ($sum\_sh < 10$)

Because we already have the continuous variable *sum_sh* which can represent the segment of a driver in a week, we also delete this column from the data set.

If a driver is not online within a week, he/she is definitely a churn driver because a driver cannot complete any trip without being active on the app. Therefore, there may exist a case that a driver does not have enough 21 rows representing information of 21 weeks (from August 1st to December 19th). He/She may churn in between and then come back again. To address this issue, we have to fill in 0 for all the attributes in the missing weeks. This action expands the data set from 761,103 rows to 986,830 rows. Figure 4.2 illustrates this step clearly.

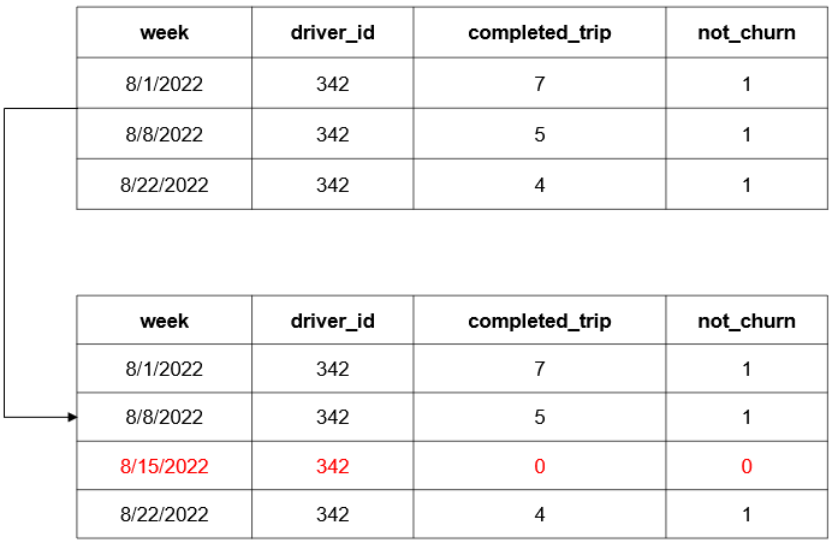

Figure 4.2: Illustration of filling 0 for all the attributes of a driver in churning weeks

After that the target column *not_churn_1w* is created by shifting forward the churn status value by 1 week. Note that week December 12th is the last week which has the future churn status value.

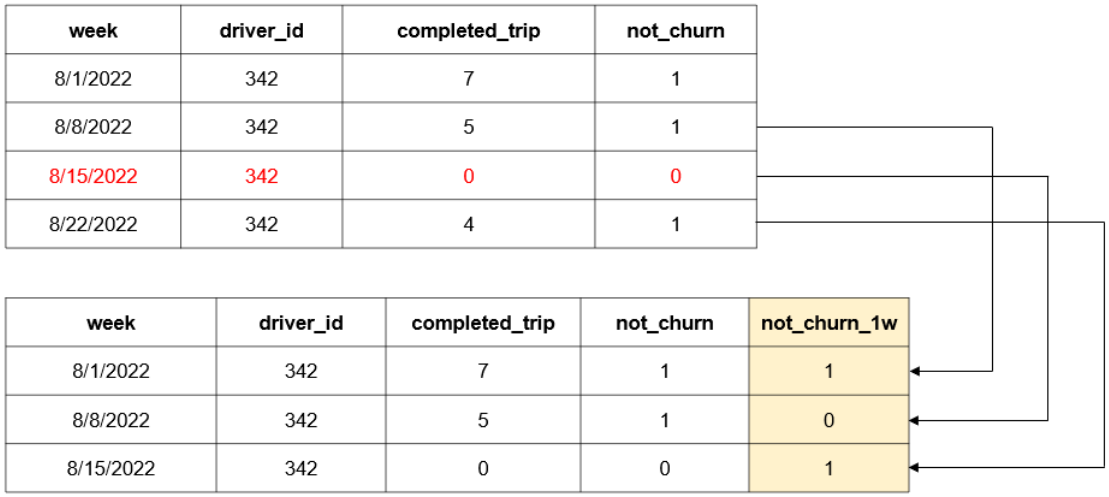

Figure 4.3: Illustration of how the target column *not_churn_1w* is created

The number of drivers fluctuates between 28,791 and 44,372 throughout time. For simplicity, I choose a time period for feature computation of $w = 1$ week and $v = 1$ for label calculation. The earliest time slice begins its predicting on August 8th. When the data is segmented into various time slices, each of which is shifted by a different driver, there are 20 alternative forecasting origins that may be utilized for training or testing (see Figure 4.4). By reserving the most recent time slice for testing and ensuring that the training and testing label windows do not overlap, a training set may include no more than 19 time slices.

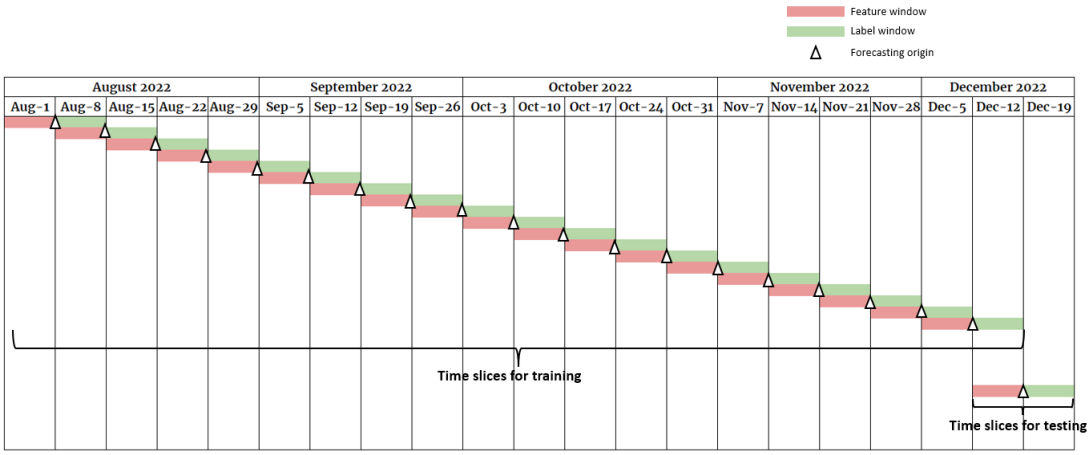

Figure 4.4: Available data horizon with time slices ($w = 1$ and $v = 1$) for training and testing

Applying multiple time slicing makes this data preprocessing step more complicated. For $w = 2$ or $w = 3$, to capture the performance of a driver in one time span, I take the average of each attribute (James Han, 2018) in that time slice. This procedure is illustrated by Figure 4.5.

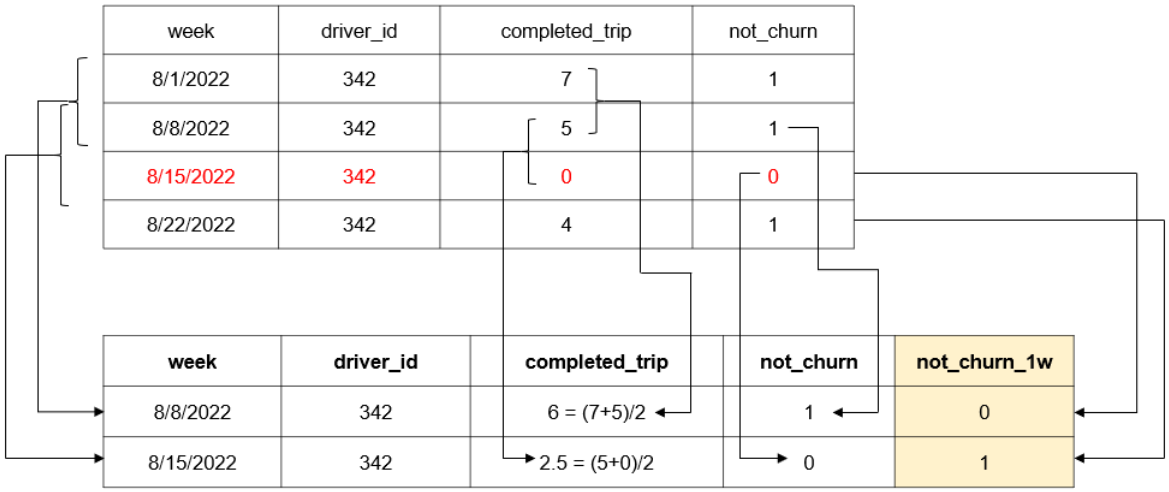

Figure 4.5: Data processing applying multiple time slicing method ($w = 2$)

Similar technique is applied when $w = 3$. Figure 4.6 shows the available data horizon with time slices $w = 3$ for training and testing.

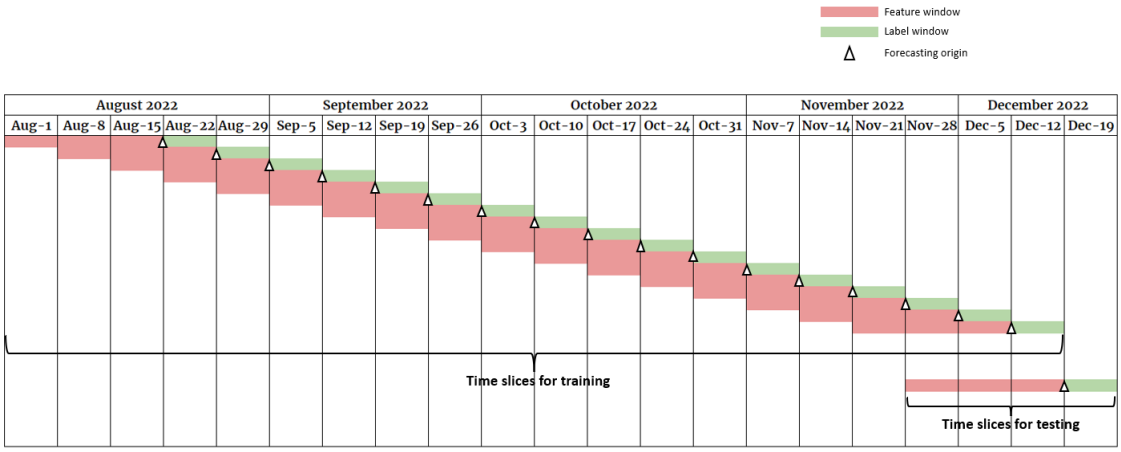

Figure 4.6: Available data horizon with time slices $w = 3$ for training and testing

4.3 Exploratory Data Analysis

Figure 4.7 demonstrates that the company's weekly churn rate fluctuated during the given period of time and exhibits seasonality. A fluctuating churn rate shows that circumstances and behavior patterns also fluctuate over time, making out-of-period testing with a time shift between the training set and the test set crucial. Incorporating circumstances from numerous time slices prevents overfitting to a single point in time, which is another reason to train on many time slices.

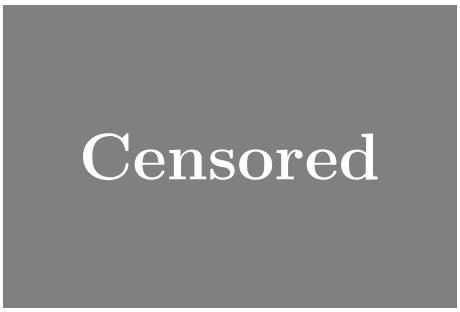

Figure 4.7: Weekly churn rate from August 1st to December 19th

4.4 Feature Selection Using WOE and IV

Calculating WOE and IV for discrete variables and continuous variables is divided into 2 different parts.

To calculate WOE and IV for continuous variables, we follow the following steps:

1. Bin the continuous variable into several categories or intervals.
2. For each bin, calculate the proportion of good (not churn) and bad (churn) observations.
3. Calculate the WOE for each bin as the logarithm of the ratio of good-to-bad proportions: WOE = ln(good proportion/bad proportion).
4. Calculate the IV as the sum of the difference between the good and bad proportions weighted by the WOE of each bin: $IV = \sum(\text{good proportion} - \text{bad proportion}) \times WOE$.
5. A higher IV indicates a more predictive variable, with values greater than 0.3 or 0.5 generally considered as strong predictors.

Features and their corresponding *IV* and *adjusted IV* of 3 ways of preprocessing data (w = 1, 2, 3) are shown below:

	DISCRETE_COL	DISTINCT	IV	IV_ADJUSTED
0	segment	5	2.120044	2.120044
1	device_type	2	0.095691	0.095691
2	vehicle_model	345	inf	0.041401
3	vehicle_brand	37	inf	0.017755
4	ops_city_id	2	0.000049	0.000049

Figure 4.8: Discrete features and their corresponding IV when $w = 1$ (slicing 1 week)

In case of simple single-slicing (w=1), *segment* is the only significant discrete feature whose *adjusted IV* = 2.12. The other four features have a relatively low

adjusted IV, then they will not be chosen to put in the Logistic regression and Random Forest model.

	CONTINUOUS_COL	BEST_CUT	IV	IV_ADJUSTED
0	sum_sh	19	inf	2.24402
1	completed_trip	19	inf	1.831873
2	Trips_without_surge	19	inf	1.811293
3	GB_without_surge	19	inf	1.429108
4	gmv_aftertax	19	inf	1.410044
5	total_fare	19	inf	1.406498
6	accepted	19	inf	1.39997
7	Valid_dispatch	19	inf	1.341706
8	driver_trip_amount	19	inf	1.305912
9	Trips_with_surge	19	inf	1.205379
10	total_ETA	19	inf	1.134872
11	distance_travelled	19	inf	1.093188
12	total_dispatched_request	19	inf	0.924153
13	Surge_Organic_GB	19	inf	0.729066
14	loyalty_amount	19	inf	0.509916
15	Surge_Contri_GB	19	inf	0.430911
16	weekly_amount	19	inf	0.239899

Figure 4.9: 16 continuous features with the highest corresponding adjusted IV when $w = 1$ (slicing 1 week)

Figure 4.9 shows that all the continuous features in the provided data set are exploratory variables that have strong predictive power with *adjusted IV* > 0.3. However, since there are some variables having a high correlation with others in the data, we need to select suitable features not only based on their high *adjusted IV* but also their low correlation with other selected features.

	DISCRETE_COL	DISTINCT	IV	IV_ADJUSTED
0	not_churn	2	2.163353	2.163353
1	segment	5	2.044522	2.044522

Figure 4.10: Discrete features and their corresponding IV when $w = 2$ (slicing 2 weeks)

When using multiple time slicing with $w = 2$, the churn status of the current observed week, represented by the variable *not_churn* is taken into consideration to calculate adjusted IV for discrete variables. Surprisingly, *not_churn* has a higher IV than *segment*, which can conclude that the churn status of next week relies much on that of the current week.(see Figure 4.10). For continuous variables, their associated adjusted IV is shown below:

	CONTINUOUS_COL	BEST_CUT	IV	IV_ADJUSTED
0	sum_sh	19	inf	2.203581
1	completed_trip	19	inf	1.810433
2	Trips_without_surge	19	inf	1.703545
3	accepted	19	inf	1.494095
4	Valid_dispatch	19	inf	1.494094
5	GB_without_surge	19	inf	1.365397
6	gmv_aftertax	19	inf	1.353995
7	total_fare	19	inf	1.350977
8	total_ETA	19	inf	1.338007
9	driver_trip_amount	19	inf	1.264073
10	Trips_with_surge	19	inf	1.232569
11	distance_travelled	19	inf	1.165097
12	total_dispatched_request	19	inf	0.845884
13	Surge_Organic_GB	19	inf	0.738391
14	loyalty_amount	19	inf	0.486083
15	Surge_Contri_GB	19	inf	0.420364

Figure 4.11: 15 continuous features with the highest corresponding adjusted IV when $w = 2$ (slicing 2 weeks)

	DISCRETE_COL	DISTINCT	IV	IV_ADJUSTED
0	not_churn	2	2.203158	2.203158
1	segment	5	1.962822	1.962822
2	not_churn_pre_1w	2	1.39795	1.39795

Figure 4.12: Discrete features and their corresponding adjusted IV when $w = 3$ (slicing 3 weeks)

Figure 4.12 shows that the churn status of the previous week also has a strong relationship to the churn flag of next week, but the level of effect is not as strong as churn status of the current week.

	CONTINUOUS_COL	BEST_CUT	IV	IV_ADJUSTED
0	sum_sh	19	inf	2.142974
1	completed_trip	19	inf	1.73273
2	Trips_without_surge	19	inf	1.670286
3	Valid_dispatch	19	inf	1.507115
4	accepted	19	inf	1.487102
5	total_ETA	19	inf	1.423983
6	GB_without_surge	19	inf	1.323688
7	gmv_aftertax	19	inf	1.317614
8	total_fare	19	inf	1.315509
9	driver_trip_amount	19	inf	1.238771
10	Trips_with_surge	19	inf	1.222946
11	distance_travelled	19	inf	1.100818
12	trip_time_hour	19	inf	0.925418
13	total_dispatched_request	19	inf	0.793484
14	Surge_Organic_GB	19	inf	0.709885

Figure 4.13: 14 continuous features with the highest corresponding adjusted IV when $w = 3$ (slicing 3 weeks)

With three ways of preprocessing data by changing the number of time slice, we observe that *sum_sh* is a feature which has the strongest power to classify the *not_churn_1w* status, followed by the information related to the number of completed trips per week and weekly drivers' earning.

After determining some good discrete and continuous predictors, we created dummies for them before applying Logistic regression and Random Forests. (see Figure 4.14)

	segment:Fulltime	segment:High Fulltime	segment:Low medium	segment:Medium	completed_trip_discrete:0 - 7	completed_trip_discrete:14 - 28	completed_trip_discrete:28 - 50
0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0
2	0	0	0	1	0	0	1
3	0	0	0	1	0	0	0
4	0	0	0	0	1	0	0
...	...	...	...	...	...	...	...
863594	0	0	0	1	0	0	1
863595	0	0	0	0	1	0	0
863596	0	0	0	0	1	0	0
863597	0	0	0	0	0	0	0
863598	0	0	1	0	1	0	0

863599 rows × 20 columns

Figure 4.14: A part of dataframe after creating dummies for significant discrete and continuous variables

4.5 Result

This section provides model estimation for Logistic regression applied for three different ways of preprocessing data, in which the number of time slice w is changed from 1 to 2 and 3 respectively. After that, the performance of individual models (including Logistic Regression and Random Forest) and hybrid models (including K-Means and these two machine learning models) will also be discussed.

4.5.1 Model Estimation

The coefficients and intercepts of three Logistic regression models applied on three dataframes created by choosing three different numbers of time slices are displayed as follows.

Feature name	**Coefficients**	**p_values**
Intercept	2.262505	NaN
segment:Fulltime	1.395465	0.00E+00
segment:High Fulltime	1.649717	0.00E+00
segment:Low medium	0.467186	2.37E-293
segment:Medium	0.914049	0.00E+00
completed_trip_discrete:0 - 7	-0.589207	3.81E-33
completed_trip_discrete:14 - 28	-0.16791	3.99E-04
completed_trip_discrete:28 - 50	-0.099736	2.83E-02
completed_trip_discrete:50 - 100	0.211041	2.17E-07
completed_trip_discrete:7 - 14	-0.298678	5.29E-10
total_fare_discrete:1 - 5 mil	0.134527	2.61E-05
total_fare_discrete:100k - 1 mil	-0.237508	4.35E-11
total_fare_discrete:<100k	-0.895514	4.59E-123
total_ETA_discrete:1 - 60 mins	-0.374421	4.02E-18
total_ETA_discrete:60 - 480 mins	-0.090636	2.55E-02
total_ETA_discrete:<1 min	-0.884775	3.84E-81
distance_travelled_discrete:1 - 10km	-0.644595	3.83E-111
distance_travelled_discrete:10 - 50km	-0.448401	6.50E-79
distance_travelled_discrete:100 - 1000km	-0.121426	4.74E-10
distance_travelled_discrete:50 - 100km	-0.242573	2.36E-24
distance_travelled_discrete:<1 km	-1.381473	0.00E+00

Table 4.1: Significant features and its coefficients after applying Logistic regression for the case of slicing 1 week

Feature name	**Coefficients**	**p_values**
Intercept	2.044741	NaN
segment:Full-time	1.52731	0.00E+00
segment:High Full-time	1.874481	0.00E+00
segment:Low-medium	0.493656	0.00E+00
segment:Medium	0.978009	0.00E+00
not_churn:0.0	-1.282574	0.00E+00
completed_trip_discrete:0 - 7	-0.474571	1.52E-24
completed_trip_discrete:14 - 28	-0.19378	1.31E-05
completed_trip_discrete:28 - 50	-0.157096	2.21E-04
completed_trip_discrete:50 - 100	0.042355	2.66E-01
completed_trip_discrete:7 - 14	-0.290586	1.17E-10
total_fare_discrete:1 - 5 mil	0.105003	4.72E-04
total_fare_discrete:100k - 1 mil	-0.172981	3.39E-07
total_fare_discrete:<100k	-0.608867	1.47E-63
total_ETA_discrete:1 - 60 mins	-0.266239	4.64E-11
total_ETA_discrete:60 - 480 mins	-0.051594	1.71E-01
total_ETA_discrete:<1 min	-0.429491	2.65E-21
distance_travelled_discrete:1 - 10km	-0.607685	4.16E-103
distance_travelled_discrete:10 - 50km	-0.401627	3.37E-67
distance_travelled_discrete:100 - 1000km	-0.12973	1.36E-12
distance_travelled_discrete:50 - 100km	-0.249156	1.01E-27
distance_travelled_discrete:<1 km	-1.300426	0.00E+00

Table 4.2: Significant features and its coefficients after applying Logistic regression for the case of slicing 2 weeks

Feature name	**Coefficients**	**p_values**
Intercept	2.001277	NaN
segment:Full-time	1.49E+00	0.00E+00
segment:High Full-time	1.86E+00	0.00E+00
segment:Low-medium	4.40E-01	1.89E-270
segment:Medium	9.00E-01	0.00E+00
not_churn:0.0	-1.63E+00	0.00E+00
not_churn_pre_1w:0.0	-4.14E-01	0.00E+00
completed_trip_discrete:0 - 7	-2.91E-01	3.76E-10
completed_trip_discrete:14 - 28	-0.088065	4.67E-02
completed_trip_discrete:28 - 50	-9.12E-02	3.10E-02
completed_trip_discrete:50 - 100	1.24E-01	1.23E-03
completed_trip_discrete:7 - 14	-1.49E-01	9.15E-04
total_fare_discrete:1 - 5 mil	1.27E-02	6.68E-01
total_fare_discrete:100k - 1 mil	-2.01E-01	1.70E-09
total_fare_discrete:<100k	-5.22E-01	2.36E-47
total_ETA_discrete:1 - 60 mins	-2.29E-01	2.13E-08
total_ETA_discrete:60 - 480 mins	-4.41E-02	2.45E-01
total_ETA_discrete:<1 min	-0.35646	3.71E-14
distance_travelled_discrete:1 - 10km	-4.88E-01	1.81E-62
distance_travelled_discrete:10 - 50km	-3.16E-01	3.43E-41
distance_travelled_discrete:100 - 1000km	-1.09E-01	1.44E-09
distance_travelled_discrete:50 - 100km	-1.54E-01	1.72E-11
distance_travelled_discrete:<1 km	-1.150856	1.44E-268

Table 4.3: Significant features and its coefficients after applying Logistic regression for the case of slicing 3 weeks

In Logistic regression, the coefficients represent the relationship between each feature (predictor) and the response (outcome) variable. A positive coefficient means that as the value of the corresponding feature increases, the log odds of the outcome also increase. In other words, a positive coefficient indicates that a higher value of

the feature is associated with a higher probability of a positive outcome. In this thesis, positive outcome means *not_churn_1w* event = 1.

On the other hand, a negative coefficient means that as the value of the corresponding feature increases, the log odds of the positive outcome decrease. In other words, a negative coefficient indicates that a higher value of the feature is associated with a lower probability of a positive outcome.

For example, table (4.1, 4.2, 4.3) shows that if a driver's segment is low medium, medium, full-time, or high full-time, he/she is more likely to stay and continue working with this ride-hailing company since the coefficients of these features are all greater than 0. On the contrary, if the number of completed trips per week is less than 50, or the weekly fare is relatively low, from 1 to 5 million dong, he/she is more likely to churn next week as the coefficients of these attributes are all negative.

4.5.2 Model Validation

Comparing the results between the training set and the testing set after running regression models or machine learning models is important for several reasons:

1. **Overfitting:** One of the biggest issues in machine learning is overfitting, where a model becomes too complex and starts to fit the noise in the training data instead of the underlying patterns. By comparing the performance of the model on the training set and the testing set, we can get an idea of whether your model is overfitting or not. If the model is overfitting, we will see a big gap between the performance on the training set and the testing set.

2. **Model generalization**: The goal of a machine learning model is to generalize well to new, unseen data. By evaluating the performance of the model on the testing set, we can get an estimate of how well it will perform on new, unseen data. If the model performs well on the training set but poorly on the testing set, it may indicate that the model is not generalizing well.

3. **Unbiased evaluation**: When we evaluate a model on the same data that was used for training, there is a risk of overstating the performance of the model. By using a separate testing set, we can get a more accurate and unbiased evaluation of the model's performance.

Framework	Cluster	Number of time slice	Precision	Recall	F1 score	AUC	Gini	KS
Individual Models								
Logistic Regression		1	0.75	0.78	0.76	0.8976	0.7953	0.6694
		2	0.78	0.76	0.77	0.9055	0.8109	0.6776
		3	0.79	0.77	0.78	0.9076	0.8151	0.6809
Random Forest		1	0.74	0.79	0.77	0.8981	0.7962	0.6701
		2	0.77	0.79	0.78	0.9065	0.8131	0.6814
		3	0.77	0.81	0.79	0.9091	0.8182	0.6871
Hybrid Models								
K-Means + Logistic Regression	1	1	0.77	0.77	0.77	0.8599	0.7197	0.6067
		2	0.34	0.13	0.19	0.6564	0.3128	0.2133
		3	0.42	0.23	0.3	0.6911	0.3823	0.2605
K-Means + Logistic Regression	2	1	0	0	0	0.5377	0.0754	0.0884
		2	0.81	0.74	0.77	0.8765	0.7529	0.6241
		3	0.3	0.08	0.13	0.5953	0.1905	0.1469
K-Means + Logistic Regression	3	1	0	0	0	0.625	0.245	0.1788
		2	0	0	0	0.5692	0.1383	0.1113
		3	0.79	0.79	0.79	0.8808	0.7616	0.6318
K-Means + Random Forest	1	1	0.77	0.77	0.77	0.8603	0.7207	0.6075
		2	0.32	0.13	0.19	0.6625	0.3249	0.2203
		3	0.47	0.22	0.3	0.6964	0.3928	0.2708
K-Means + Random Forest	2	1	0	0	0	0.5379	0.759	0.0884
		2	0.77	0.81	0.79	0.8776	0.7552	0.6273
		3	0.33	0.08	0.13	0.6003	0.2007	0.1557
K-Means + Random Forest	3	1	0.57	0	0	0.6326	0.2652	0.1881
		2	0	0	0	0.5734	0.1468	0.1191
		3	0.77	0.83	0.8	0.8828	0.7656	0.6355

Figure 4.15: Model performance of the training set

Framework	Cluster	Number of time slice	Precision	Recall	F1 score	AUC	Gini	KS
Individual Models								
Logistic Regression		1	0.8	0.8	0.8	0.9037	0.8073	0.6913
		2	0.81	0.8	0.81	0.9107	0.8214	0.6974
		3	0.81	0.81	0.81	0.9119	0.8237	0.6955
Random Forest		1	0.8	0.8	0.8	0.9033	0.8067	0.6917
		2	0.8	0.82	0.81	0.9111	0.8223	0.6993
		3	0.81	0.81	0.81	0.9128	0.8257	0.6984
Hybrid Models								
K-Means + Logistic Regression	1	1	0.8	0.8	0.8	0.8993	0.7985	0.6911
		2	0.82	0.78	0.8	0.8919	0.7838	0.6847
		3	0.82	0.78	0.8	0.8743	0.7486	0.6847
K-Means + Logistic Regression	2	1	0	0	0	0.7538	0.5075	0.5157
		2	0.81	0.8	0.8	0.9058	0.8116	0.6947
		3	0.81	0.79	0.8	0.8852	0.7703	0.6846
K-Means + Logistic Regression	3	1	0.62	0.94	0.75	0.7431	0.4862	0.6142
		2	0.81	0.79	0.8	0.8777	0.7554	0.6846
		3	0.83	0.76	0.8	0.9081	0.8161	0.6936
K-Means + Random Forest	1	1	0.8	0.8	0.8	0.9005	0.8011	0.6916
		2	0.76	0.81	0.79	0.8322	0.6644	0.6707
		3	0.44	0.1	0.16	0.82	0.6401	0.645
K-Means + Random Forest	2	1	0	0	0	0.7579	0.5158	0.5532
		2	0.79	0.82	0.81	0.9022	0.8044	0.698
		3	0.84	0.72	0.77	0.8346	0.6692	0.6398
K-Means + Random Forest	3	1	0.21	0.02	0.03	0.6286	0.2573	0.509
		2	0.83	0.75	0.79	0.8504	0.7008	0.6657
		3	0.81	0.8	0.81	0.9069	0.8138	0.6965

Figure 4.16: Model performance of the testing set

Overall, for individual models, there is no big gap between the performance of training and testing set, indicating that my driver churn prediction models do not have over-fitting or under-fitting problems.

Regarding the performance of individual models, Random Forest applied on data whose number of time slice equals to 3 turned out to be the most effective model to predict churn drivers next week based on the results from not only training set but also testing set, with F1 score is 0.78 and 0.81 respectively. This can conclude that using longer time slices may help increase the performance of the driver churn prediction model slightly, by around 1%.

Considering AUC and Gini index for Logistic regression and Random Forest model, both AUC and Gini index are all greater than 0.8 for the testing set. A perfect model will have AUC = 1 and a completely random classifier has AUC = 0.5. Usually, a good model score has AUC ranging somewhere in between. The figure of AUC for Random Forest with $w = 3$ is 0.9128, meaning that there is around 91% that the model will be able to distinguish between positive (not churn) class and

negative (churn) class. Its Gini index is 0.8257, indicating that the model has a high capacity for discriminating between two classes.

To assess the performance of both individual and hybrid models, we compare the evaluation metrics of individual models to the average metrics of the hybrid models, calculated from the metrics of each cluster. As seen in Table 4.15, all hybrid methods have lower performance compared to the individual models and may have an overfitting problem since there is a big gap between the evaluation metrics of the training set and the testing set. For instance, the F1 score for the combination of K-Means and Logistic regression of cluster 1 on the testing set is 0.8, doubling that of the training set. It is noticeable that for each cluster, there is only one model has good results with a specific number of time slice. For cluster 1, the combination of K-Means + Logistic regression and K-Means + Random Forest performs the most efficiently on data with $w = 1$. For cluster 2, the result of K-Means + Logistic Regression and K-Means + Random Forest is considered as good with $w = 2$. Data with $w = 3$ helps the model generated by the combination of K-Means + Logistic regression and K-Means + Random Forest combinations reached such good performance.

The results of training and testing set yield two meaningful insights into building driver churn prediction models for ride-hailing companies. Firstly, multiple time slicing does have a positive effect on improving the performance of the model. It is important to determine the number of time slices w before applying any regression or machine learning models. The higher number of time slices is, the better the performance of Logistic regression and Random Forest model becomes. Secondly, using hybrid models to predict churn drivers for this Vietnamese ride-hailing company was not as efficient as we expected. The reasons behind have not yet been investigated but it might raise a topic for further research.

Chapter 5

DISCUSSION

In this thesis, I conducted to build driver churn prediction models following two different approaches, which are (1) using only one model (Logistic or Random Forest) and (2) using hybrid models (formed by the segmentation stage and the stage of applying regression/machine learning models on each group). Besides, I evaluated the difference between training on multiple time slices and training on a single time slice. The results show that training on multiple time slices is more efficient and leads to better performance compared to training on just one time slice. My study compared single-slicing and different versions of multi-slicing to determine the impact of these methods on the model's performance. The results indicate that choosing a longer time slice contributes to improved prediction performance.

However, there may exist some limitations in this thesis that need to be researched in the future. Firstly, although ([39]) shows that hybrid models show better results than individual models in predicting commercial customers' credit scores, when it comes to driver churn prediction models, all the evaluation metrics of hybrid models combined by K-Means and Logistic regression or K-Means and Random Forest were relatively worse than individual models. Secondly, with hybrid models, there exist overfitting problems since there is a significant difference between the result of the training and testing set.

These findings have intriguing implications for ride-hailing companies attempt-

ing to anticipate drivers' churn status in the near future. To increase the predictive potential of the driver churn prediction model, choosing a longer time slice should be considered since this will take the churn history of drivers into account, resulting in better prediction accuracy. Another suggestion will be to develop driver churn prediction models using other regressions machine learning models such as XGBOOST, Gradient Boosting, LSTM (Long Short-term Model), etc. Moreover, future studies should construct hybrid models with other clustering techniques instead of K-Means.

Chapter 6

APPENDIX

6.1 Logistic Regression And Random Forest Code With $w = 1$

Automatically generated by Colaboratory.

Original file is located at

https://drive.google.com/file/d/10WbFceSQuS4_9CGrs8IuyASihsw1vnHb/view?usp=sharing

---

6.2 Logistic Regression And Random Forest Code With $w = 2$

Automatically generated by Colaboratory.

Original file is located at

https://drive.google.com/file/d/18PbIArg7CP1vBMt_xEdcUOKT9fxhIhIz/view?usp=sharing

---

6.3 Logistic Regression And Random Forest Code With $w = 3$

Automatically generated by Colaboratory.

```
Original file is located at
https://drive.google.com/file/d/1firUB-qYb_xxTXJU9kpCF7DRyhnSPr6O/view?usp=sharing
```

6.4 Code With $w = 1$ Using Hybrid Model

```
Automatically generated by Colaboratory.
Original file is located at
https://drive.google.com/file/d/1BL7xLBZoRuU4vfdRLdj7JqawbfLza4xX/view?usp=sharing
```

6.5 Code With $w = 2$ Using Hybrid Model

```
Automatically generated by Colaboratory.
Original file is located at
https://drive.google.com/file/d/1DF7so-TNwPKRGqHg8uyaeX81msm5gbWJ/view?usp=sharing
```

6.6 Code With $w = 3$ Using Hybrid Model

```
Automatically generated by Colaboratory.
Original file is located at
https://drive.google.com/file/d/19PYHdYpf-ZEd629MH7zk9RcIL3sClr6R/view?usp=sharing
```

6.7 Data Attributes And Description

Variable's name	Data type	Description	Num_distinct_values	Num_missing_values	Example
week	datetime64	Observed week	21	0	01/08/2022
driver_id	int64	Driver ID	62,237	0	342
activated_date	datetime64	Date when driver's account is verified	1,114	0	20/07/2019
ops_city_id	int64	City where driver lives (189 = HCMC, 190 = Hanoi)	2	0	189
device_type	int64	Driver's smartphone type (1 = Android, 0 = iOS)	2	0	0
ftt_time	datetime64	Date when driver completed the first trip	1,381	0	10/12/2018
vehicle_brand	string	Driver's vehicle brand	53	0	Toyota
vehicle_model	string	Driver's vehicle model	377	65	Avanza
sum_sh	float64	Total online hours in the observed week of a driver	238,133	0	4.46
segment	string	Driver segment (High full-time, Full-time, Medium, Low medium, Part-time) based on total completed trips in the observed week	5	0	Part-time
completed_trip	float64	Number of completed trips in the observed week of a driver	292	29,057	7
total_dispatched_request	float64	Number of booking requests in the observed week of a driver	1,307	29,057	12
Valid_dispatch	float64	Number of valid booking requests in the observed week of a driver	509	29,057	9
accepted	float64	Number of accepted requests by driver in the observed week	459	29,057	9
cancelled	float64	Number of cancelled requests of a driver in the observed week	360	29,057	2
rider_cancel	float64	Number of trips cancelled by riders after a request is sent to a driver in the observed week	225	29,057	0
driver_cancel	float64	Number of trips cancelled by driver in the observed week	371	29,057	2
total_fare	float64	Total fare of completed trips of a driver in the observed week (VND)	66,993	29,057	845,000
gmv_aftertax	float64	Total fare after tax of completed trips of a driver in the observed week (VND)	596,034	29,057	797,745
Trips_without_surge	float64	Number of trips without surge of a driver in the observed week	248	29,057	6
Trips_with_surge	float64	Number of trips with surge of a driver in the observed week	140	29,057	1
GB_without_surge	float64	Total fare of completed trips without surge of a driver in the observed week (VND)	60,676	29,057	750,000
Surge_Organic_GB	float64	Total organic fare of completed trips with surge of a driver in the observed week (VND)	421,380	186,621	63,333.3
Surge_Contri_GB	float64	Total surge fare of completed trips with surge of a driver in the observed week (VND)	429,337	186,621	31,666.7
total_ETA	float64	Total estimated time arrival of a driver in the observed week (minute)	1,966	50,895	52
total_ATA	float64	Total actual time arrival of a driver in the observed week (minute)	1,884	76,363	80
trip_time_hour	float64	Total time a driver drives from the pick-up venue to the destination in the observed week (hour)	3,862	75,883	3
en_route_hour	float64	Total time a driver drivers from his/her location to the pick-up venue in the observed week (hour)	1,885	75,875	1
distance_travelled	float64	Total travelled distance of a driver in the observed week (km)	88,978	29,057	90
driver_trip_amount	float64	Total amount that a driver actually earns in the observed week (VND)	243,133	29,057	525,163
weekly_amount	float64	Total weekly incentive amount that a driver earns in the observed week (VND)	12,738	476,958	110,000
loyalty_amount	float64	Total incentive amount that a driver earns from the loyalty program in the observed week (VND)	51,174	74,786	402,345
not_churn	int64	Churn status of the observed week of a driver (1 = not churn, 0 = churn)	2	0	1

Figure 6.1: Data Attributes And Description

Bibliography

[1] Risselada, H., Verhoef, P. C., & Bijmolt, T. H. (2010). *Staying power of churn prediction models.* Journal of Interactive Marketing, 24(3), 198–208.

[2] Burez, J., & Van den Poel, D. (2007). *CRM at a pay-TV company: Using analytical models to reduce customer attrition by targeted marketing for subscription services.* Expert Systems with Applications, 32(2), 277–288.

[3] Burez, J., & Van den Poel, D. (2009). *Handling class imbalance in customer churn prediction.* Expert Systems with Applications, 36(3), 4626–4636.

[4] Gordini, N., & Veglio, V. (2014). *Customer relationship management and data mining: A classification decision tree to predict customer purchasing behavior in global market.* In P. M. Vasant (Ed.), Handbook of research on novel soft computing intelligent algorithms: Theory and practical applications. I, . IGI Global. http://dx.doi.org/10.4018/978-1-4666-4450-2.

[5] Lemmens, A., & Croux, C. (2006). *Bagging and boosting classification trees to predict churn.* Journal of Marketing Research, 43(2), 276–286. http://dx.doi.org/10.1509/ jmkr.43.2.276.

[6] Levin, N., & Zahavi, J. (2001). *Predictive modeling using segmentation.* Journal of Interactive Marketing, 15(2), 2–22. http://dx.doi.org/10.1002/dir.1007.

[7] Michael C Mozer, Richard Wolniewicz, David B Grimes, Eric Johnson, Howard Kaushansky, and Athene Software, *Churn Reduction in the Wireless Industry.*, In Advances in Neural Information Processing Systems, volume 12, pages 935–941, 2000.

[8] Neslin, S. a., Gupta, S., Kamakura, W., Lu, J., & Mason, C. H. (2006). *Detection : Defection measuring of the predictive accuracy understanding models churn customer.* Journal of Marketing Research, 43(2), 204–211.

[9] Risselada, H., Verhoef, P. C., & Bijmolt, T. H. A. (2010). *Staying power of churn prediction models.* Journal of Interactive Marketing, 24(3), 198–208. http://dx.doi.org/10.1016/j. intmar.2010.04.002.

[10] Verbeke, W., Martens, D., Mues, C., & Baesens, B. (2011). Building comprehensible customer churn prediction models with advanced rule induction techniques. Expert Systems with Applications, 38(3), 2354–2364. http://dx.doi.org/10.1016/j.eswa.2010. 08.023.

[11] Verbeke, W., Dejaeger, K., Martens, D., Hur, J., & Baesens, B. (2012). New insights into churn prediction in the telecommunication sector: A profit driven data mining approach. European Journal of Operational Research, 218(1), 211–229. http://dx.doi.org/10.1016/j.ejor.2011.09.031.

[12] Au, W. H., Chan, K. C. C., & Yao, X. (2003). A novel evolutionary data mining algorithm with application to churn prediction. IEEE Transactions on Evolutionary Computation, 7, 532–545.

[13] Gordini, N. (2013). Genetich algorithms for small enterprises default prediction: Empirical evidence from Italy. In P. M. Vasant (Ed.), Handbook of research on Novel Soft Computing intelligent algorithms: Theory and practical applications. I. (pp. 258–293) IGI Global. (doi:10.4018/978-1-4666-4450-2).

[14] Buckinx, W., & Van Den Poel, D. (2005). *Customer base analysis: Partial defection of behaviourally loyal clients in a non-contractual FMCG retail setting. European* Journal of Operational Research, 164(1), 252–268. http://dx.doi.org/10.1016/j.ejor. 2003.12.010.

[15] Sharma, A., & Kumar Panigrahi, P. (2011). *A neural network based approach for predicting customer churn in cellular network services.* International Journal of Computer Applications, 27(11), 26–31. http://dx.doi.org/10.5120/3344-4605.

[16] Coussementt, K. (2014). *Improving customer retention management through cost-sensitive learning.* European Journal of Marketing, 48(3/4), 477–495. http://dx.doi.org/10.1108/EJM-03-2012-0180.

[17] Larivière, B., & Van den Poel, D. (2004). *Investigating the role of product features in preventing customer churn, by using survival analysis and choice modeling: The case of financial services.* Expert Systems with Applications, 27(2), 277–285. http://dx.doi.org/10.1016/j.eswa.2004.02.002.

[18] Xie, Y., Li, X., Ngai, E. W. T., & Ying, W. (2009). *Customer churn prediction using improved balanced Random Forests. Expert Systems with Applications*, 36, 5445–5449. http://dx. doi.org/10.1016/j.eswa.2008.06.121.

[19] Brij Masand, Piew Datta, D. R. Mani, and Bin Li. CHAMP, *A prototype for automated cellular churn prediction.*, Data Mining and Knowledge Discovery, 3(2):219–225, 1999.

[20] Qiannan Zhu et al 2019 IOP Conf. Ser.: Mater. Sci. Eng. 631 052008

[21] TAH Tran, KP Thai, *Rider Churn Prediction Model for Ride-Hailing Service: A Machine Learning Approach*, International Conference on Artificial Intelligence and Big Data in Digital Era, pages 301-314, 2022.

[22] James Han, *Churn Prediction.*
`https://itsjameshan.github.io/Churn-rate-for-Uber/churn2.nb.html`

[23] Enioluwamo Ireoluwa Obatoki, *Predicting driver churn with Logistic regression: A case-study of drivers in the ride-hailing industry*, Journal of Emerging Technologies and Innovative Research (JETIR), 2019.

[24] Siddiqi, N. *Credit Risk Scorecards, Developing and Implementing Intelligent Credit Scoring*, Hoboken, NJ: John Wiley & Sons, Inc., 2006.

[25] C. Brooks, *Introductory Econometrics for Finance*, Cambridge University Press, 2019.

[26] Gür Ali and Arıtürk (2014): *Dynamic churn prediction framework with more effective use of rare event data: The case of private banking*. Expert systems with applications, 41(17), 7889-7903.

[27] Seppälä, T., Thuy, L. (2018). A combination of multi-period training data and ensemble methods to improve churn classification of housing loan customers. In Proceedings of the 2nd international conference on advanced research methods and analytics (CARMA 2018) (pp. 141–144). Universidad Politècnica de València.

[28] Leung, H. C., Chung, W. (2020). A Dynamic Classification Approach to Churn Prediction in Banking Industry. In Amcis 2020 proceedings data science and analytics for decision support. Association for Information Systems

[29] Arno De Caignya, Kristof coussementt, Koen W. De Bock (2018), *A new hybrid classification algorithm for customer churn prediction based on Logistic regression and decision trees*, European Journal of Operational Research, 269 (2018), 760–772.

[30] Weiss, G. M. (2004). Mining with rarity: A unifying framework. SIGKDD Explorations, 6(1), 7–19.

[31] E. Zdravevski, P. Lameski and A. Kulakov, "Weight of evidence as a tool for attribute transformation in the preprocessing stage of supervised learning algorithms," The 2011 International Joint Conference on Neural Networks, San Jose, CA, USA, 2011, pp. 181-188, doi: 10.1109/IJCNN.2011.6033219.

[32] Dalia Atif, Mabrouka Salmi. 2022. Feature Selection for Credit Risk Classification. Intelligent Systems and Pattern Recognition, pages 165-179.

[33] R. Anderson, "The Credit Scoring Toolkit - Theory and Practice for Retail Credit Risk Management and Decision Automation", Oxford University Press Inc., New York, 2007

[34] Lidia Ceriani and Paolo Verme, via ResearchGate. "The Origins of the Gini Index: Extracts from Variabilità e Mutabilità (1912) by Corrado Gini," The Journal of Economic Inequality, September 2012, vol. 10, no. 3, Pages 1–23.

[35] Egan, J. P. (1975). Signal detection theory and ROC-analysis. In Series in cognition and perception. Academic press.

[36] Thomas L. C. Crook J. N. Edelman D. B. Society for Industrial and Applied Mathematics. (2002). *Credit scoring and its applications*. Society for Industrial and Applied Mathematics.

[37] Coussement, K., Lessmann, S., Verstraeten, G. (2017). A comparative analysis of data preparation algorithms for customer churn prediction: A case study in the telecommunication industry. Decision Support Systems, 95, 27–36.

[38] Verma, Prashant (2020), *Churn Prediction for Savings Bank Customers: A Machine Learning Approach*, Journal of Statistics Applications Probability: Vol. 9: Iss. 3, Article 10. DOI: http://dx.doi.org/10.18576/jsap/090310

[39] Marcos Roberto Machado, Salma Karray, 2022 *Assessing credit risk of commercial customers using hybrid machine learning algorithms*, Expert Systems with Applications, Volume 200,2022,116889,ISSN 0957-4174, https://doi.org/10.1016/j.eswa.2022.116889.

[40] L. Breiman. Bagging Predictoors. Machine Learning, 24(2), 123–140, 1996.