國立臺灣大學社會科學院經濟學系

碩士論文

Department of Economics
College of Social Sciences
National Taiwan University

Master Thesis

使用機器學習技法預測消費者的購買行為：以網站的點擊資料為例

Predicting Consumers' Purchase Decision by Clickstream Data: A Machine Learning Approach

陳伯駒
Po Chu Chen

指導教授：
林明仁 博士

Advisors:
Ming-Jen Lin, Ph.D.

中華民國 107 年 1 月
Jan, 2018

國立臺灣大學碩士學位論文
口試委員會審定書

使用機器學習技法預測消費者的購買行為：以網站的點擊資料為例

Predicting Consumers' Purchase Decision by Clickstream Data: A Machine Learning Approach

本論文係陳伯駒君 (R04323050) 在國立臺灣大學經濟學系完成之碩士學位論文，於民國 107 年 1 月 31 日承下列考試委員審查通過及口試及格，特此證明

口試委員：

_____

_____

_____ _____

謝辭

此篇論文的完成，雖然不是什麼曠世鉅作，卻也花了不少時間！尤其是做個體實證後，才發現原來清理與處理資料才是佔用最多時間的地方，無奈的是這部分沒有辦法寫入論文中，因此首先需要特別感謝李宗穎老師與王子豪學長耐心地帶著我了解處理資料的程序，還有陳釗而老師給予我許多機器學習的相關資源與意見，如果沒有他們我想我沒有辦法如此有效率地把資料清理乾淨！當然更要感謝資料的提供者、也就是我的指導教授林明仁老師，老師聰敏的思緒以及獨到的見解正是吸引我想要接受老師指導的原因，每次跟老師討論完都會覺得又學到了一些新東西與想法、不管是資料處理的技巧或學術研究裡的眉角，都讓我收穫滿滿。在這漫漫的研究路途上，雖然辛苦卻也交到了許多志同道合的好朋友！感謝一同在 653 與 648 奮鬥的好戰友們，大家在數學課的疲勞轟炸下一起互相鼓勵、討論研究問題，讓我倍感溫馨、也為煩悶的心情增添一絲絲的喜悅。謝謝永遠的好戰友兼飯友浩軒、尚傑在我口試前幫了我許多忙、讓我可以專心準備口試的內容！以及郁婷熱情地跟我分享關於論文寫作與口試的經驗，當然還有瑄華姐姐及湘羚姐姐的各種正能量宣傳與捧場，和願意當我的心情垃圾桶。另外還要感謝去年狂找我取暖的筱筑，在國外狂歡之於也不忘在口試前替我加油打氣、共同組成崩潰取暖團。另外謝謝詩媛熱心地提供我

latex 模板，幫助我在排版方面沒有太大的進入障礙！還有世希幫忙我做壓力與焦慮的釋放、以及口試前耐著性子聽完我將近一小時有如天書的報告。還有一位在 Berkeley 交換時結交到的韓國好友智賢，謝謝你願意在求職之際還幫我檢查文法與錯字，真的是台韓友好！然而在這一路上的點點滴滴，受惠於人之眾多、族繁不及備載，無法將所有細節都納入謝辭，在最後的感謝就是我的父母親，謝謝爸爸媽媽一路以來的支持與照顧、讓任性的我能夠繼續留在台大求學與使用資源，沒有你們就沒有現在的我、當然就沒有現在這篇論文！謝謝你們！

摘要

近年來，許多商業型態逐漸由實體商店轉型成網路商店，即我們所熟知的電子商務。隨著網路運算的進步，這些網路平台儲存了鉅量的訪客登入及瀏覽資訊，亦稱為點擊資料。在本研究中，我們主要分析一家線上酒店零售商的網站點擊資料、使用兩種常用的機器學習模型：決策樹與隨機森林，預測消費者最終的購買行為。除了引入消費者在網站上的搜尋特徵，我們另外建立了一種根據消費者之登入頁面順序、進行訪客分類的分群結果，並利用此分群結果作為統整性的特徵納入學習模型。經過重複採樣消除非平衡數據的問題後，我們兩個最終的學習模型都達到高於 90% 的整體預測率，並且提供了廠商未來可能進一步行銷的訪客類型。

關鍵詞：點擊資料、決策樹、隨機森林、電子商務。

Abstract

In the recent years, numerous commerces have gradually shifted from physical store to web-shops, so-called the e-commerce. These online stores contain lots of log files in the back-end which basically record the pages accessed by visitors, namely the clickstream data. In this study, we predict consumers' purchase decision by analyzing the clickstream data from an online wine retailer. We impose two modern machine learning model, decision tree and random forest, to predict consumers' final purchase intention. Besides the normal features based on visitors' activities on the website, we construct a new feature that clusters different groups of visitors according to the sequence page-type accessed. After re-sampling to remedy the unbalanced data, our two models both show high predictive accuracy up to 90% and provides a new insight for retailer to target some specific visitors on website.

Keywords: Clickstream Data, Decision Tree, Random Forest, E-Commerce

Contents

List of Figures

List of Tables

Chapter 1

Introduction

E-commerce has become one of the most popular platform for retailers in the recent years, people in nowadays are more willing to visit the e-commerce website than the actual physical stores, these e-commerce websites thus play an important role in business. The online market provides several advantages for merchants compared to physical stores with regard to wide visibility, market area and relatively low cost of operating. Despite of the associative inconvenience, such as anonymity of visitors, accompanied with this trend, it is still worthy to extract the potential customers concealed in these anonymous visitors. On the contrary to physical stores, web-stores have a lack of direct observability of visitors' intentions and preferences. Thanks to the developments of web technologies, it becomes much easier for users to store abundant of records as log file on the server, which provides more insights into visitors' behaviors, so-called the clickstream data. Due to the anonymity and complexity in the online environment, obtaining the accurate and comprehensive perception of visitors' online behaviors is a critical issue for merchants' success.

Generally, clickstream data contains information about visitors' behaviors on the website, such as the pages viewed, figures clicked or products purchased. The data in this paper is basically visitors' browsing log file from a wine retailer's web-store. Log files punctu-

ally record the events occurring in an operating system, so many detailed information of users' identity and action can be found in the log files. We construct a new identification of browsing history based on $667,068$ unique visitors and use these identifications as an observation level. The advantage of using browsing history as observation level is that the predictive result could be directly applied by the firms and also correspond to our main question: what kind of search pattern can predict the purchase decision accurately? In this research, we first cluster these browsing history by the type of pages they viewed. The package **TraMineR** in R provides an intuitive and convenient algorithm which is able to cluster the sequences of states, and the states we used in this paper is exact the type of viewed pages. The clustering result can be considered as different group of searching behaviors, so it will also become one of the major features for our prediction models.

The formal prediction models we implement are two modern machine learning models: decision tree and random forest, they are especially suitable for the highly non-polynomial model just like our scenario and the results are easy to interpret. Also, the algorithms are efficient to impose in R environment due to the established **CARET** packages. Besides, our data exists the imbalanced problem[1] , we are going to impose several re-sampling methods to remedy this issue and eventually select the model with highest accuracy and sensitivity. Both of the prediction models perform well up to 90% accuracy and has similar sensitivity up to 90% as well.

This paper is organized as follows. We will remark the package manuals and articles related with this research in chapter 2. In chapter 3, we will make the detailed description of our data set and illustrate the information by some tables and summary statistics. Chapter 4 mainly describes the process of executing the models and tuning related parameters in algorithm. Chapter 5 provides the result of prediction and models' performance. Chapter 6 concludes the contribution and extensional works in the future.

[1] Imbalanced data sets are a special case for classification problem where the class distribution is not uniform among the classes. Typically, they are composed of two classes: The majority (negative) class and the minority (positive) class.

Chapter 2

Literature Review

Due to the development of network technology, merchants are able to provide goods or services in an online web-store. Moreover, the clickstream data collected from purchases on the web-stores can provide much more information (Moe and Fader 2004). Clickstream data generally contains users' comprehensive actions before purchases, and thus makes the opportunity of improving the understanding of visitors' activities become a critical competitive advantage (Andersen et al. 2000). Bucklin et al. (2002) also concludes that the detailed nature of information tracking about internet usage and e-commerce transactions provides an enormous opportunity for empirical modelers to enhance the prediction of choice behavior. However, the advanced environment in e-commerce may implicate some flaws. Bucklin and Sismeiro (2009) indicates that the visitors on e-commerce web-stores are rarely loyal to a particular web-store when searching a specific product. Moreover, the conversion rate[1] is generally low. Approximately 75% of the internet users browsed and searched for the specific products, but up to 65% of them never bought products by using internet (Sismeiro and Bucklin 2004). The data we obtained is highly corresponding to the scenario and property mentioned above, it consequently intrigues the main goal of our research.

[1] Conversion rate is defined by the percentage of website visits which eventually lead to a purchase.

With regard to predicting purchase intention, there are several articles have preliminary results. Rizwan (2017) simply used the linear regression in machine learning model to derive the predictive result. On the other hand, Verheijden (2012) used the logistic regression to obtain visitors' probability of purchasing on the website. For the classification model, Hop and van de Velden (2013) implemented logistic, random forest, Support Vector Machine (SVM) and neural network to predict the purchase intention and evaluate each model's performance. The results shows random forest has a highest predictive accuracy, implying the non-polynomial model could be more suitable for the data in web-store environment. Besides, e Costa Magalhães Teixeira (2015) followed both of the random forest and logistic regression. Random forest eventually obtained a higher accuracy as well. According to the above articles, there are two possible improvements which we can work on: First, the features extracted in above articles are basically based on visitors' action on the website, but we also want to capture the order of their browsing action. For instance, a visitor first clicks the product pages then clicks the advertising pages and a visitor clicks the advertising then clicks product pages may be considered as two extinct browsing action, or may be a similar action compared to other browsing action. Namely, we would like to distinguish the extinct clusters of visitors based on the order of their clicks. Second, their data did not have a severely imbalanced problem, which is contrary to our data. As a result, we need to refer to multiple re-sampling methods to deal with this problem (Menardi and Torelli 2014).

In fact, research about clustering the visitors on website has been widely completed. Most of them implemented the popular k-means method[2] to partition the visitors into k groups (Kumar and Guruprasad 2015). However, in order to take the order of browsing action into accounts, we actually follow the technique from Gabadinho et al. (2011) which applied the similar centroid-methods to cluster the sequences of state with the package **TraMineR** in R.

[2] k-means clustering aims to partition n observations into pre-defined k clusters in which each observation belongs to the cluster with the nearest mean, serving as a centroid of the cluster.

Theoretically, there are two basic re-sampling methods: under-sampling and over-sampling. The debut between these two methods always exists. For example, Drummond et al. (2003) concluded that the under-sampling can actually beat over-sampling. In reality though, we should not merely apply under- or over-sampling on our dataset. We need to account for the cross validation and evaluate the final performance. Moreover, there are hybrid methods which combine under-sampling with generation of additional data: **ROSE** (Lunardon et al. 2014) and **SMOTE** (Chawla et al. 2002). We will totally refer to above four re-sampling methods to manipulate the imbalanced problem.

Chapter 3

Data Processing

Our data comes from an online wine retailer whose website contains a bunch of visitors' browsing log file, the duration of our data is from January 6, 2015 to January 1, 2017, and there are $667,068$ unique visitors among this duration. The method that we identify a unique visitor is combining their IP address, user-agent and browser cookie. User-agent and browser cookie can be considered as the information of browser used by the visitors. For example, two visitors may have a same IP address and both use Chrome to visit the website, but if they log in Chrome with different google account, the user-agent or cookie will be different, consequently viewed as two unique visitors.

Every time the visitor clicks a link on website, page log file will be recorded, and the total viewed pages are $4,983,067$. We then categorize the type of these pages, such as "main_wine" is the homepage on website; "mem_login" means this visitor is in member login page, "wine_list" is mainly the page of searching result from this visitors, and if the visitor clicks a specific product from the result page, the "wine_detail" page will pop out, other detailed types are shown in Table 1.

Table 1: Composition of Page Type

Page Type	Frequency	Percent	Cum
list_wine	1,798,236	36.09	36.09
wine_detail	1,217,020	24.42	60.51
main_wine	445,982	8.95	69.46
winery_intro	322,926	6.48	75.94
edm	236,998	4.76	80.70
spirit_detail	218,460	4.38	85.08
wine_cart1	129,780	2.60	87.68
wine_cart1_coupon	110,717	2.22	89.91
list_spirit	107,641	2.16	92.07
vipepaper	84,661	1.70	93.77
mem_login	39,264	0.79	94.55
wine_recommend	26,527	0.53	95.09
mem_member	24,832	0.50	95.58
main_spirit	22,488	0.45	96.04
...	...	...	...
spirit_variety	70	0.00	100.00
spirit_partial_pop	31	0.00	100.00
spirit_recommend	25	0.00	100.00
Total	4,983,067	100.00	

3.1 Visitors' Browsing History

Although we initially identify the unique visitors by creating their visitor-id, we also create another identification called "history-id" which is similar to visitor-id. History-id is created by capturing the browsing history before each visitors' sent orders; on the other hand, if the visitor did not send any order in his whole visiting session, history-id will capture his whole browsing record. Namely, history-id can be considered as the identification of each browsing history which may send an order or not. In our formal model, we will mainly use history-id as an observation level since it fits our main goal. If we use visitor-id instead, we may underestimate the effect from the visitors whose browsing history had sent multiple orders[1] . Therefore, it should be reminded that history-id is the

[1]Imaging there are two visitors, one had sent 1 order in his visiting session; the other had sent 5 orders. If we use visitor-id as an observation level, both of them are the "order-sent" visitors. However, our main purpose is to predict the classification of sending order based on the browsing behaviors, it is more ap-

observation level in our model.

There are totally 672,756 history-id among these unique visitors, and 6,583 of them had sent an order at the end, namely, only 1% of the browsing record had sent an order. The imbalanced data is a common issue in e-commerce since most visitors may just skim through the website, or even find their desirable products but choose to purchase from other web stores. There are several ways to deal with the imbalanced data, we will impose four re-sampling methods to remedy the problem. Table 2 shows an overview of the order conversion for history-id, note that "register state" represents whether this visitor is registered as a member before he sent an order, we can observe that 50% of visitors with membership had sent an order at the end, which is intuitive as well since the registered visitors can be treated as "serious" buyer controversial to "casual" browsers.

Table 2: Order Conversion among History Record

	Register State		Total
	no	yes	
Order Sent			
no	664,068	2,105	666,173
yes	4,320	2,263	6,583
Total	668,388	4,368	672,756

3.2 Overview of Browsing Behavior

In order to be familiar with the features of browsing history, we need to inspect some brief information of visitors' browsing behaviors. Checking visitors' time spent on website and number of pages they viewed is pretty straightforward. The summary statistics is shown in Table 3, we can find that the distribution of "duration" and "pages" are both extremely skewed to right, representing there exists amounts of inactive visitors who had merely

propriate to extract 5 browsing behaviors before order sent instead of merely applying the whole browsing history as 1 observation. The main reason is that applying only 1 observation may lose some information specific to each order-sent decision, such as personal demand or seasonality. Thus, we will totally create 6 observations in above scenario.

visited the website for few times.

Note that if a visitor only visits the website once, then his duration will be 0 since the log file does not record a leaving out action; however, this would not significantly affect our analysis since we may put more effort on other active visitors[2]. Due to above characteristic, we will conduct some rules to deal with these users and evade some redundant computation, the detailed rules will be covered in later section.

Table 3: Browsing Information

	Mean	S.D.	Min.	Max.	Skewness	Kurtosis	Observations
Duration (minutes)	10,290.48	1,462,931	0	8.31E+08	419.203	198,312.6	672,756
Page	5.48	57.94	1	29851	412.856	209,010.2	672,756

Note: Duration is the total time spent on website before an order sent.

Page is the total viewed pages on website before an order sent.

[2] Although we did not define the active visitors or inactive visitors, the intuitive idea of the formal definition is determined by visitors' interaction on the website, such as clicking, searching, browsing and so on.

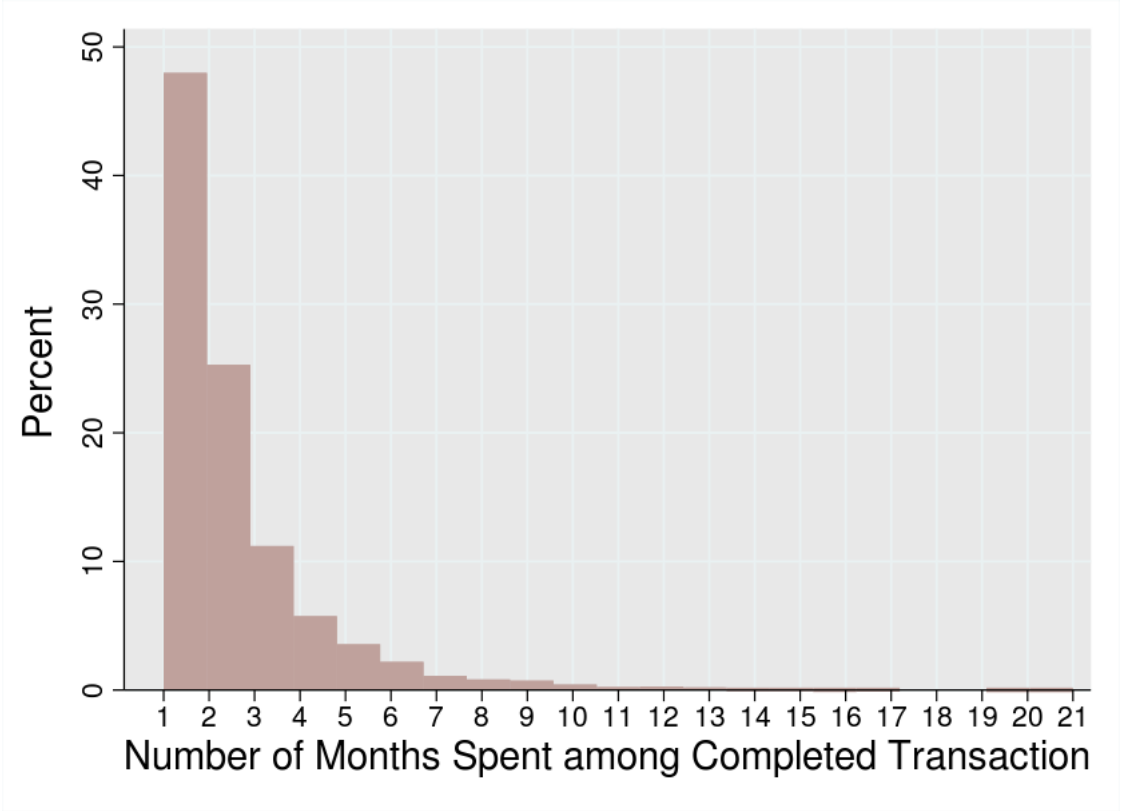
Chapter 4

Model

Although there are two formal models in this research, we also include an additional model of mining sequence data, which aims to cluster the visitors on the website according to their accessed page-type. Thus we will cover the model setting of sequence clustering, decision tree, random forest respectively in this section.

Before implementing the model, we ask ourselves a question: should we include whole browsing record for each history_id? As the previous section mentioned, there exists considerable inactive visitors during the time span. Furthermore, if we include the whole browsing record in the model to derive the result, it seems impractical for retailers since it is difficult to predict purchase decision by using an extremely long period of data. Figure 1 shows the distribution of time spent for each browsing record before order sent, most transactions are completed within two months, so we conclude that 2 months is a feasible period for us to capture. For the history_id which had sent an order, we extract its browsing records within a month before the transaction, including the transaction month as well; on the other hand, for the history_id without sending an order, we extract their browsing records for 2 months starting from the timing they entered the website. Briefly to say, both types of history_id contain the browsing records within 2 months, one starts from the beginning but the other starts from a month before the order sent.

Figure 1: Distribution of Transaction Time



4.1 Sequence Clustering

Thanks to the convenient package **TraMineR** in R, it is easy for us to cluster the visitors by their accessed-page sequence and visualize the state sequence for each group. The basic idea of the packages is first setting the state sequences for each history-id, then creating a ”distance matrix”, which is a symmetric matrix since it shows the ”distances” between each history-id. This matrix could be computed by several methods, we will use the most popular method, Longest Common Prefix(LCP) distances. At the meant time, it takes lots of time to derive the distance matrix if we did not organize these state sequences. Thus, we will use **WeightedCluster** package before imposing **TraMineR**.

In **WeightedCluster** package, the identical sequences will be aggregated, it is useful especially for a large number of data since it saves memory and computing time. Besides, in order to create the distance matrix, the largest matrix that can be stored in R is 2^{31}, which means the number of cells cannot exceed it. If we impose each history record to

the algorithm without cleaning, the distance matrix will crush R eventually. Hence, we should do some cleaning works as the rules[1] before executing the algorithms.

First of all, we checked the distribution of viewed pages for order-sent history_id, and the result is shown in Table 4 and Figure 2 respectively. We can observe their minimum viewed page is 2, implying those history_id which had merely entered the website once does not provide any further information, so we first drop the history_id with only 1 record. Secondly, most of the viewed pages in Figure 2 are blow 1000 pages, implying the viewed pages exceeding 1000 pages does not provide extra information as well. Hence, we drop the viewed pages exceeding 1000 pages. Briefly to say, we clean the browsing records by dropping the viewed pages less than 1 or larger than 1000.

Table 4: Summary of Viewed Pages for Order-Sent History_id

Variable	Mean	Std. Dev.	min	Max	N
Number of Viewed Pages	83.59	108.01	2	1,592	6,583

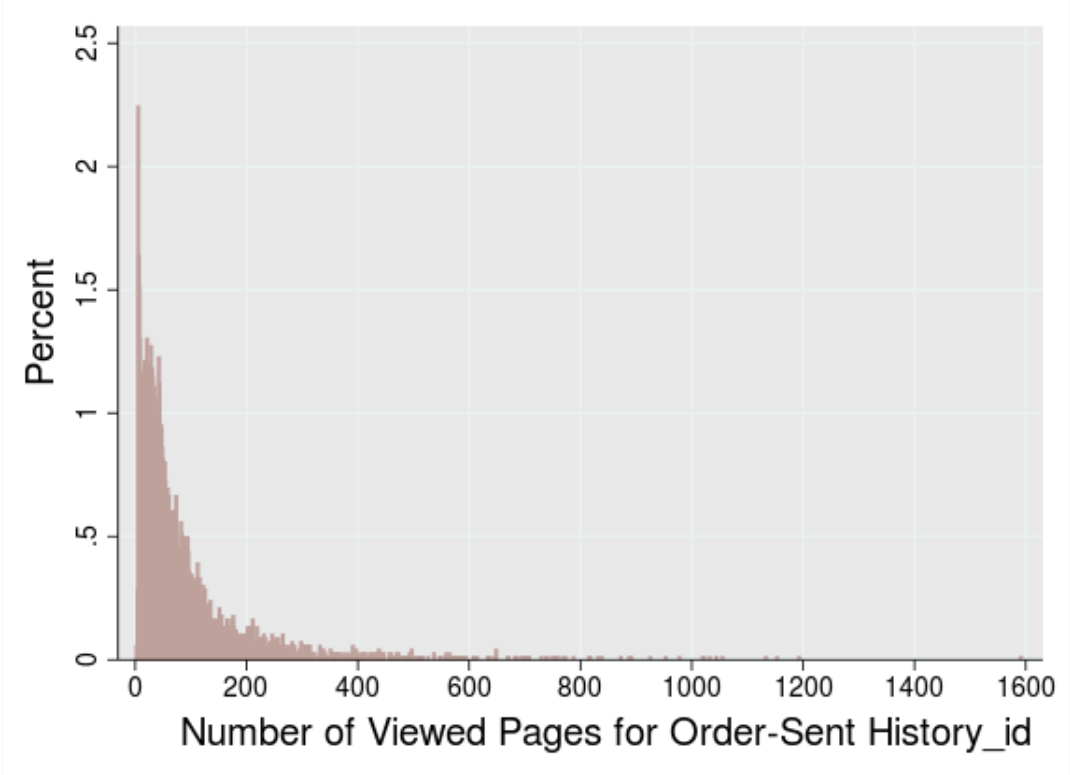
Furthermore, we initially attempted to include whole page-types as the state sequences in weighted clustering algorithm; however, the problem as mentioned happened unfortunately due to the limited size of a distance matrix. In order to reduce the matrix size, we recheck the page-type composition in Table 1 and find it is mostly composed of the type ”list_wine”, ”wine_detail” and ”main_wine”. As a result, we alternatively select them as the representative states of sequences.

4.2 Decision Tree

A decision tree is constructed by repeatedly splitting the observation into subgroups based on some features, and the features we extract from data are DURATION, CLUSTER, PAGES, MONTH and REG-STATE, where DURATION and PAGES are the time spent and number of

[1] The following rules are specific to the sequence clustering.

Figure 2: Distribution of Viewed Pages for Order-Sent History_id



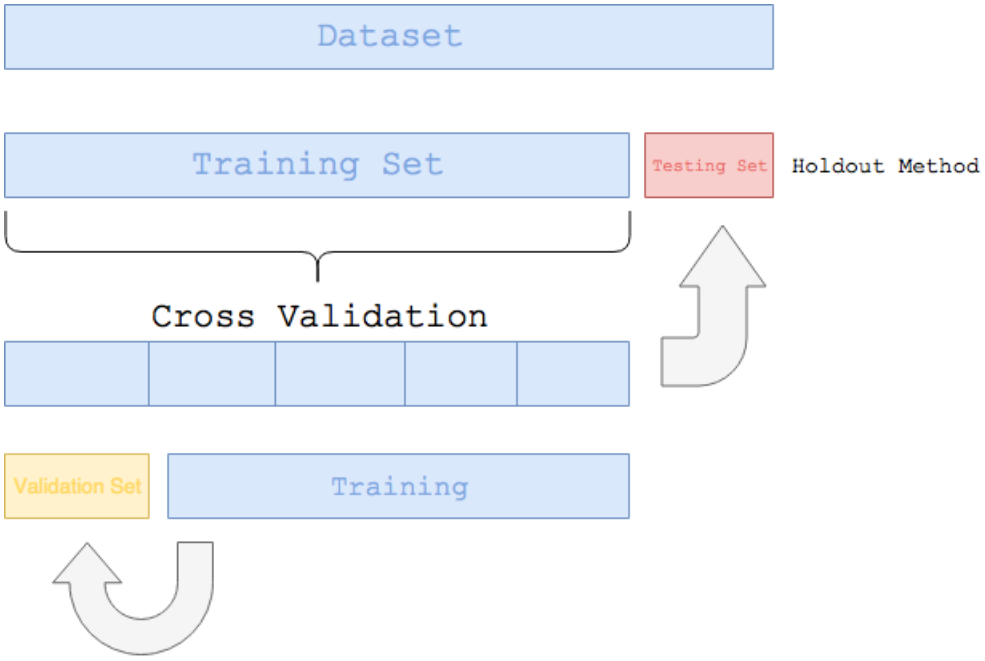
viewed pages respectively, CLUSTER is the weighted clustering result, and REG-STATE indicates whether the observation is registered as member before the order sent. MONTH captures the the month when the order is sent. The algorithm will try to find the best division on each iteration, and the feature selected in previous iteration will not be chosen again. This model is especially suitable for the classification problem based on some sort of thresholds. In fact, this is one of the main reason why we choose this model to classify the observation, since it is literally difficult to assume that the model between purchase decision and browsing behaviors is a specific polynomial form, such as linear or quadratic form. Besides, most distributions of browsing information are positive-skewed, intuitively denoting that visitors generally do not spent too much time on the website, even for the buyers. Accordingly, it is reasonable to utilize decision tree since we look forward to the result which targets some specific thresholds about visitors' "effort" on the website.

There is a hyperparameter[2] in decision tree model, complexity parameter (cp), which

[2]Hyperparameter is a parameter whose value is set before the learning algorithm executes.

can be considered as a criteria of splitting or not. The concrete rule is following: if any split does not increase the overall prediction rate by at least "cp", the algorithm will stop splitting. Namely, tree is tend to split and has more branches if cp is smaller. The optimal value of cp is decided by k-folds cross validation, and $k = 10$ is acceptable and common in practical[3]. With respect to cross validation, we first split the data for 80% as training set and 20% as testing set, then implement the cross validation on the training set. The concrete idea is shown in Figure 3 as follows.

Figure 3: Visual Representation of Train/Test Split and Cross Validation



To deal with the imbalanced problem, we conduct four re-sampling methods: over-sampling, under-sampling and the hybrid of previous two sampling, then choose the final model with the highest accuracy. With oversampling, we randomly duplicate the samples from the minority so as to match the number of sample in each class; with under-sampling, we randomly select the samples from majority so as to match the number of sample in each class. Regarding to hybrid methods, the packages **ROSE** and **SMOTE** are the most commonly used. **ROSE** uses smooth bootstrap to create the artificial samples from the feature space neighborhood around the minority class; on the other hand, **SMOTE** draws artifi-

[3] In fact, 5-Fold or 10-Fold generally works well.

cial samples by choosing points that lie on the line connecting the rare observation to one of its nearest neighbors in the feature space.

4.3 Random Forest

As its name, random forest is constructed by a large number of decision trees, each tree is trained on the bootstrapped sample from training set, and each split in the tree is based on the random subsets of feature space. The prediction is obtained by majority voting among these decision trees. With this randomness, random forest is able to improve models' stability and accuracy compared to purely 1 decision tree.

The hyperparameters in random forest are the number of features considered randomly at each tree split: n_{tree}, and the number of trees: m_{try}. It should be noted that n_{tree} has no local optimum, in other words, the accuracy will not increase anymore by adding trees at some point. Hence, n_{tree} can be set as an arbitrarily large value. The accuracy of random forest is affected by two factors: a higher *correlation* between any two trees in the forest is tend to increase the error rate; and a higher power(accuracy) of an individual tree in the forest tends to decrease the error rate. m_{try} affects both of above two factors: a lower value of m_{try} will decrease the correlation and predictive power; and a higher value of m_{try} will increase the correlation and predictive power. As a consequence, the selection of m_{try} is a trade-off, we need the cross validation to find the optimal m_{try} as well.

In a nutshell, the algorithm of random forest in R is executed as follows: A predetermined number of trees n_{tree} is generated. For each tree, a bootstrap sample is drawn from the training set. These trees are then trained and split only from the fixed number of randomly chosen features m_{try}. Predictions could be made by feeding a new observation to these individual trees and then making the majority vote.

Chapter 5

Result

5.1 Weighted Clustering

After applying the cleaning rules, we have $229,574$ sequences of page-type and $41,433$ unique sequences among them. We compute the distance matrix by Longest Common Prefix (LCP) method, which measures the distance/similarity according to the length of the longest common prefix. The matrix is a 41433×41433 [1]and symmetric matrix, the (i, j)-entry represents the distance between sequence i and sequence j, so the symmetric elements are all zero. Table 5 shows the first 5 rows and columns of the distance matrix. We will implement the Partitioning Around Medoids (PAM), algorithm to do the clustering, it aims to obtain the best partitioning of dataset into a predetermined k groups. Medoid is defined as the observation of a group which has the smallest weighted sum of distances from the other observations(sequences) to this group, so the algorithm seeks to minimize the distance from the medoid to other observations.

Since the k groups is predefined, we need to measure the quality of clustering among different k, and the following are the three measures we impose: "Point Biserial Correlation(PBC)", "Average Silhouette Width(ASW)" and "Hubert's C(HC)". PBC measures

[1] $41433^2 < 2^{31}$

Table 5: Illustration of Distance Matrix

Sequences	S_1	S_2	S_3	S_4	S_5	$\cdots$
S_1	0	3	73	113	700	$\cdots$
S_2	3	0	70	110	697	$\cdots$
S_3	73	70	0	40	627	$\cdots$
S_4	113	110	40	0	587	$\cdots$
S_5	700	697	627	587	0	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$

the capacity of a partition of data to regenerate the distance matrix, so the partition is valid if the value is high. Whereas the ASW measures the coherence among the partitions, high coherence represents a strong within-group homogeneity and great distance between different groups, the high ASW thus indicates the good partition of data. HC index compares the gap between the partition obtained and the distance matrix. Contrary to previous two indexes, a small value of HC indicates the good partition of dataset. We totally apply $k = 2$ to $k = 10$ and the result is shown in Figure 4. It seems $k = 3$ is a good enough solution for final clustering, we consequently implement the algorithm with predefined $k = 3$.

Figure 4: Quality of Clustering

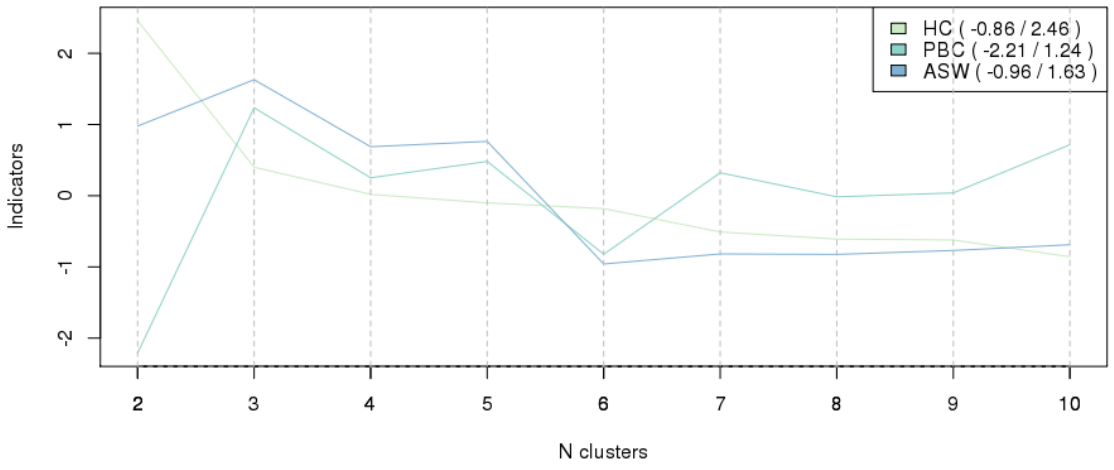
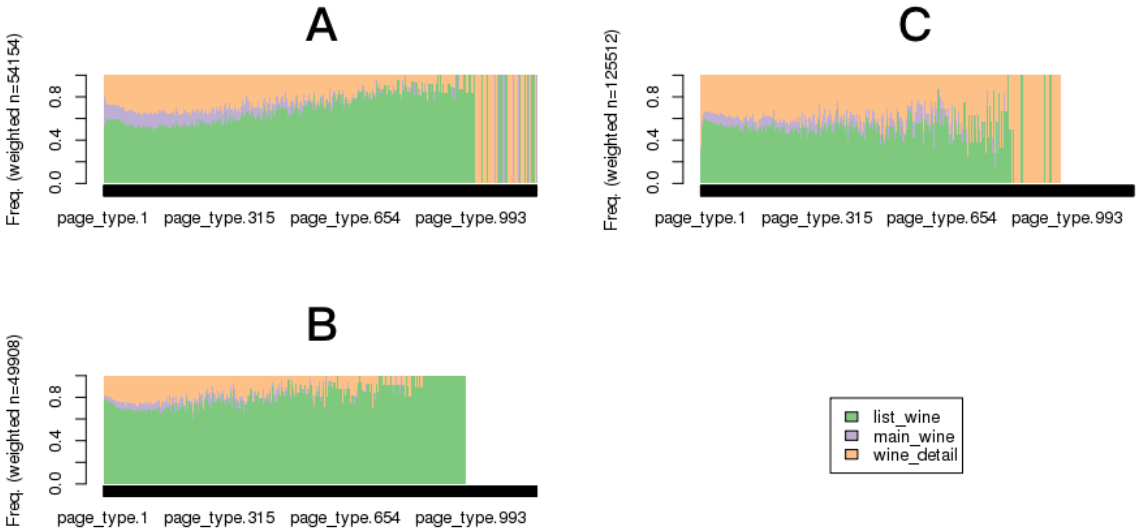


Figure 5 presents the clustering result with 3 groups. Group A contains the longest sequences of states and the largest frequency of "main_wine". Most of the page-types in group C are list_wine, so it can be considered as the group which contains many shoppers; on the other hand, group C contains the largest frequency of "wine_detail", which means the observations in group C are likely to click in and check the products compared to group B. In particular, there is a group D as well, which represents the observations dropped by our cleaning rules. Basically, the observations in group D are those history_id containing merely 1 page or containing no page-type with "main_wine", "list_wine" or "wine_detail". In brief, group D barely browses the information about products on the website.

Figure 5: Result of Clustering



5.2 Classifiers' Performance

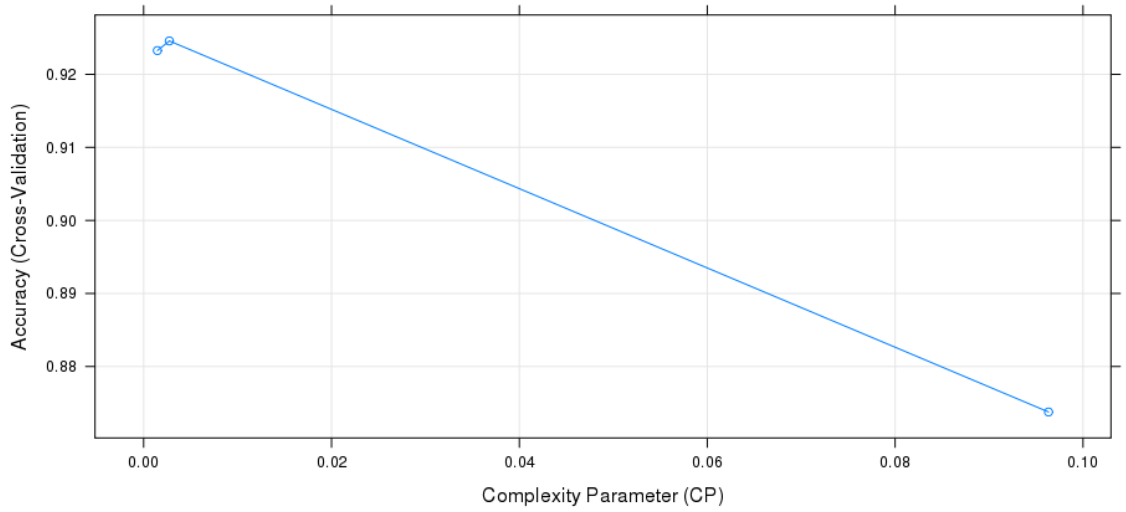
Although we have attempted four re-sampling methods to remedy the imbalanced data problem, we will mainly show the result of **SMOTE** re-sampling in this section since they have similar overall accuracy, except for the **ROSE** which has the worst performance in decision tree model and Over-Sampling which has the worst performance in random forest model. The results of other re-sampling methods are in Appendix A and Appendix B

5.2.1 Hyperparameters Tuning

Decision Tree

In previous section, we discussed about the complexity parameter cp which we need to optimize to derive the best performance. Algorithm trains the data and attempts several values of cp via 10-folds cross validation, and the process is illustrated in Figure 6. The final value of cp used for the model is 0.002736. With this choice of cp, the average estimated binary classification accuracy is 0.9245.

Figure 6: Tuning Result of Decision Tree



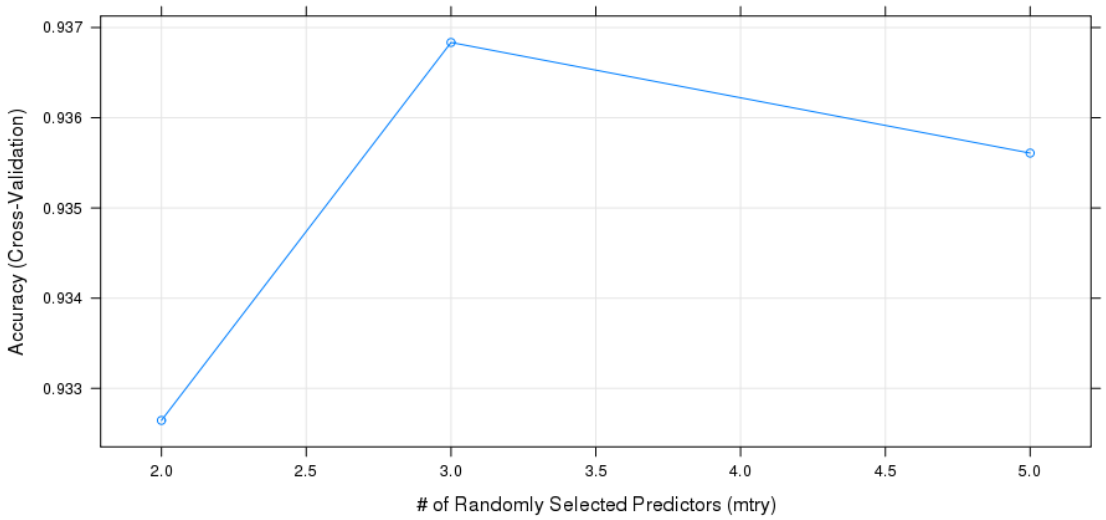
Random Forest

Despite the two predefined parameters, n_{tree} and m_{try} in random forest model, we only need to optimize the m_{try} through cross validation since the accuracy will not increase by adding trees at some particular point. In fact, n_{tree} does not lead to the over-fitting problem[2], it seems that we could set n_{tree} as large as possible. Nevertheless, the exceed number of trees will merely give rise to redundant computation. Figure 7 illustrates the process that m_{try} is attempted by algorithm, the optimal cross-validated accuracy is at-

[2]Over-fitting refers to the problem that a model predicts the training set too well. That is to say, the model is trained to learn the information and noise in the training set, which will consequently impact the performance of the model in the new data(out-of-sample).

tained at $m_{try} = 3$, and the average estimated accuracy is 0.9368. Regarding to the tuning of n_{tree}, we will show it in the later section.

Figure 7: Tuning Result of Random Forest



5.2.2 Confusion Matrix and Accuracy

Decision Tree

The performance of classification could be simply illustrated by the confusion matrix, where the diagonal elements are true-negative(TN) and true-positive(TP) respectively, other elements are false-negative(FN) and false-negative(FP) on the contrary. Accuracy is also simply calculated by $\frac{TN+TP}{N}$ where N is the total number of sample in testing set; on the other hand, the error rate(mis-classification rate) would be $\frac{FN+FP}{N}$ or directly calculated by $1 - Accuracy$. In addition, the terminology "sensitivity" is also called the true-positive rate, namely it computes how often the model predicts "yes" when the sample is actually yes[3], the formula is $\frac{TP}{\text{\# of actual yes}}$. On the other hand, "precision" measures the accuracy rate conditional on model's prediction on "yes", namely the formula is $\frac{TP}{\text{\# of predicted yes}}$. Since sensitivity and precision are both critical measure of accuracy in our scenario, we will derive a synthesized measure called "F1 score" which is computed

[3] "No" class represents the history_id did not send an order in our model; "Yes" class represents the history_id did send an order.

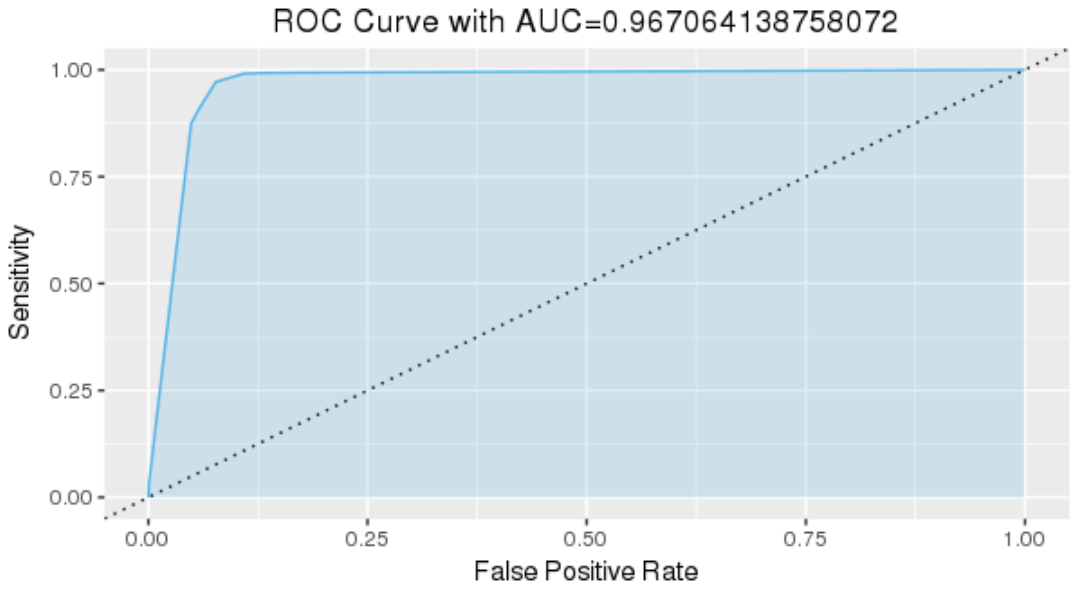
by the harmonic mean of sensitivity and precision. In this section, we will basically show the re-sampling with highest F1 score, which is the result of **SMOTE** method. Besides, there is another evaluation called ROC curve which plots the relation between sensitivity and false-positive rate (specificity)for different cutoff points of a parameter. Each point on the ROC curve represents a sensitivity-specificity pair corresponding to a specific decision threshold which is determined by the probabilities of classifying result, the top left corner of ROC curve is intuitively the point with least error rate of mis-classifying since it has larges sensitivity and lowest false-positive rate. In particular, the area under the ROC curve (AUC) is the measure of the performance that the parameter can distinguish between negative and positive.

Table 6 shows the detailed result of confusion matrix, the overall optimal accuracy is 92.35%, which is similar to the average estimated accuracy 92.45% derived from the tuning result of cross validation. This result indicates that average cross-validated accuracy could be a useful approximation for out-of-sample accuracy. Furthermore, we can assure that the average estimated accuracy is comparable across different prediction model via cross validation, the sensitivity and precision are 97.17% and 11.31% respectively, and the F1 score is 20.25% in decision tree model. Figure 8 plots the ROC curve for decision tree model with its AUC is 0.967.

Table 6: Confusion Matrix of Decision Tree

N=134,551	**Predicted: No**	**Predicted: Yes**
Actual: No	122,945 (91%)	10,260 (7.6%)
Actual: Yes	38 (0%)	1,308 (0.9%)

Figure 8: ROC curve for Decision Tree



Random Forest

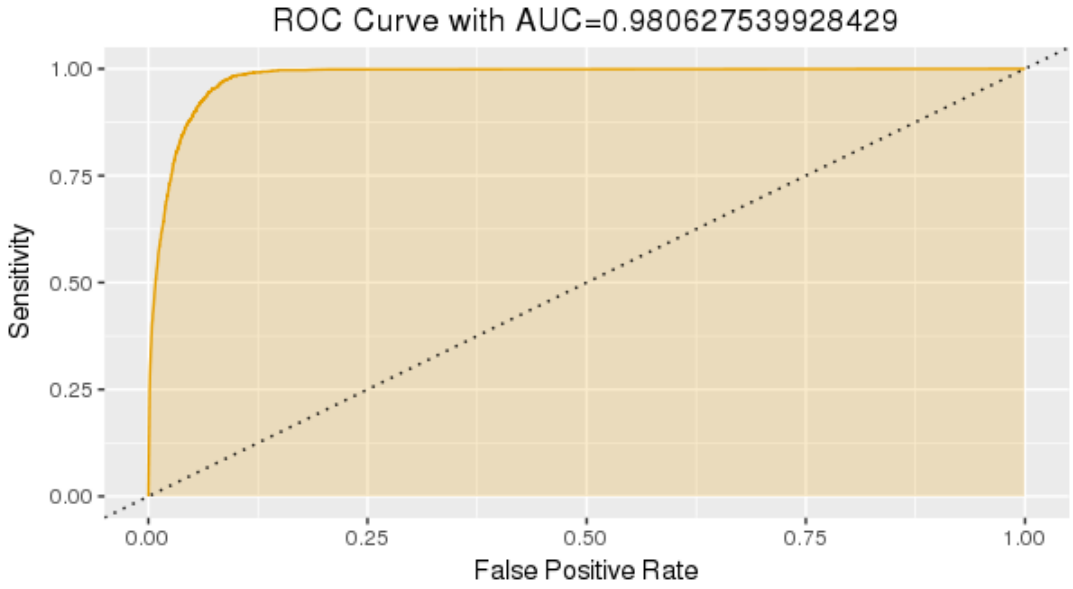
Based on above terminologies, the confusion matrix of random forest is shown in Table 7, the result is slightly different to decision tree. First, the overall optimal accuracy is 93.69% which is higher than the accuracy of decision tree model. Second, sensitivity here is 92.86% which is relatively lower than the value in decision tree. The possible reason is that the random forest is a collection of abundant classifiers, it tends to optimize the overall accuracy even under the re-sampling method, and the issue of imbalanced data itself may also lead to the problem. Nevertheless, we are apt to accept the result in random forest since decision tree may practically exist the over-fitting problem. Moreover, the precision and F1 score are 12.96% and 22.75% respectively, which are both greater than the value in decision tree model. Conclusively to say, random forest model has a better performance and stability, figure 9 shows the AUC= 0.9806 in random forest model which is higher than the value in decision tree model as well.

We have derived the accuracy of random forest, now we can finally check out how the number of trees n_{tree} affects the accuracy. Random Forest algorithm will compute the error rate in training set corresponding to n_{tree} automatically. The plot of overall error rate, error rate for the ”No” class and error rate for the ”Yes” class is shown in Figure

Table 7: Confusion Matrix of Random Forest

N=134,551	**Predicted: No**	**Predicted: Yes**
Actual: No	124,813 (92%)	8,392 (6.2 %)
Actual: Yes	96 (0%)	1,250 (0.9%)

Figure 9: ROC curve for Random Forest



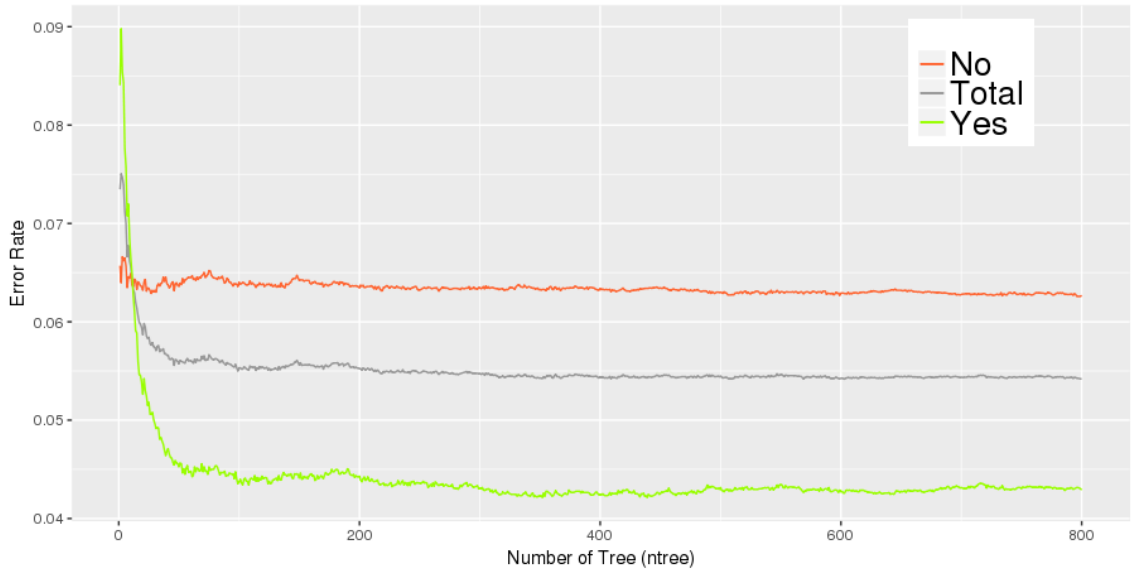
10. We can observe that the model at $n_{tree} \approx 100$ has already generated a good accuracy, but increasing the number of trees can make the error rate more stable, even it does not provide additional accuracy up to $n_{tree} \approx 500$.

5.2.3 Variable Importance

Decision Tree

The visualization of result derived by decision tree algorithm is intuitive and easy to interpret. Making the prediction from a new observation is done by starting at the root of the tree, then moving down iteratively and split according to the specific features on the branch. Eventually, this new observation will reach a single node of the tree. The illus-

Figure 10: Learning Curve of Random Forest



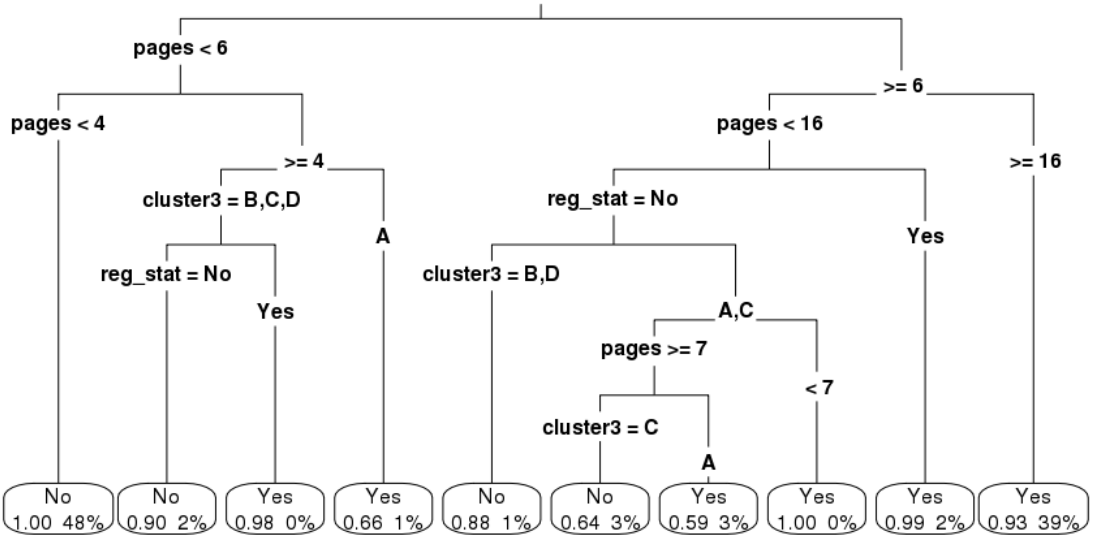
tration is visualized in Figure 11, the end of the leaf node is label with the class predicted by model, and the decimal represents the probability that the new observation will be the labeled class, whereas the percentage indicates the percent of observations in the labeled node[4]. For example, if an observation whose viewed pages are greater than 6 but lower than 16, and its registered status is a member, then the probability of this observation being classified as "Yes" is 0.99 and the observations matching above specific feature take up 2% of the whole observations. Since the feature selected in the previous iteration will not be chosen again and the split is based on the best division, the variable importance is definitely the order of splitting. For instances, we can infer the feature "PAGES" is the most important, and secondly is the "REG-STATE". Interestingly, the class "A" in CLUSTER tends to be the "Yes" class among other sequence clustering class. This result is quite intuitive since if we review the clustering result from Figure 5, class "A" contains the most viewed pages and the information of browsing products on the website.

Random Forest

Unlike the decision tree, we cannot visualize every node in random forest since the for-

[4]Note that we have utilized the re-sampling method to duplicate the minority, which is "Yes" class in our scenario. Hence, some percentages for "Yes" class may exceed the original value.

Figure 11: Visualization of Decision Tree



est contains 800 trees in our model. Figure 12 and Figure 13 show one of the sample tree within 800 trees, it is difficult for us to inspect the exact order of splitting on features since they are randomly selected dependent on m_{try}. Fortunately, the package in random forest algorithm computes two criteria to evaluate the variable importance: Mean Decrease Accuracy and Mean Decrease Gini. Mean Decrease Accuracy denotes the proportion of observations which are incorrectly classified caused by removing the specific feature; on the other hand, GINI importance evaluates the average gain of purity by splitting the observations based on the given feature. In other words, Mean Decrease Accuracy can be considered as the global variable importance, and Mean Decrease Gini can be considered as the local one. Therefore, we can simply conclude that the feature PAGES and DURATION are significant to the prediction model by Figure 13. Although the importance order of Accuracy and Gini are quite different among other three features, the order of Mean Decrease Accuracy is more acceptable practically since GINI-gain represents the local splits, which is not necessarily to measure compared with the change of overall model performance. In fact, GINI importance is relatively more biased and unstable, thus tends to answer the more indirect question.

Figure 12: Sketch of the Forest

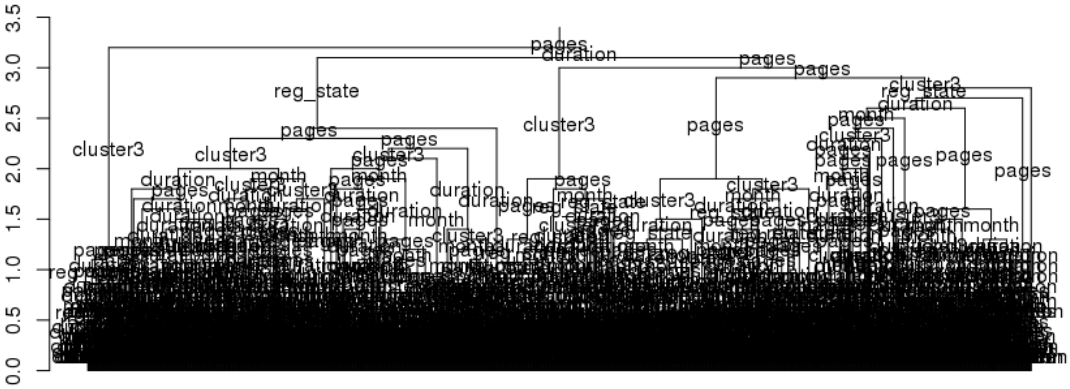


Figure 13: Sketch of the Forest with Partial Enlargement

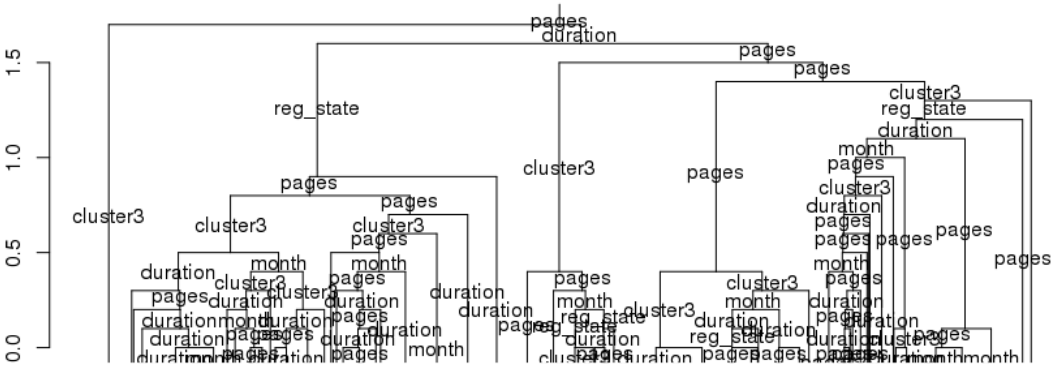
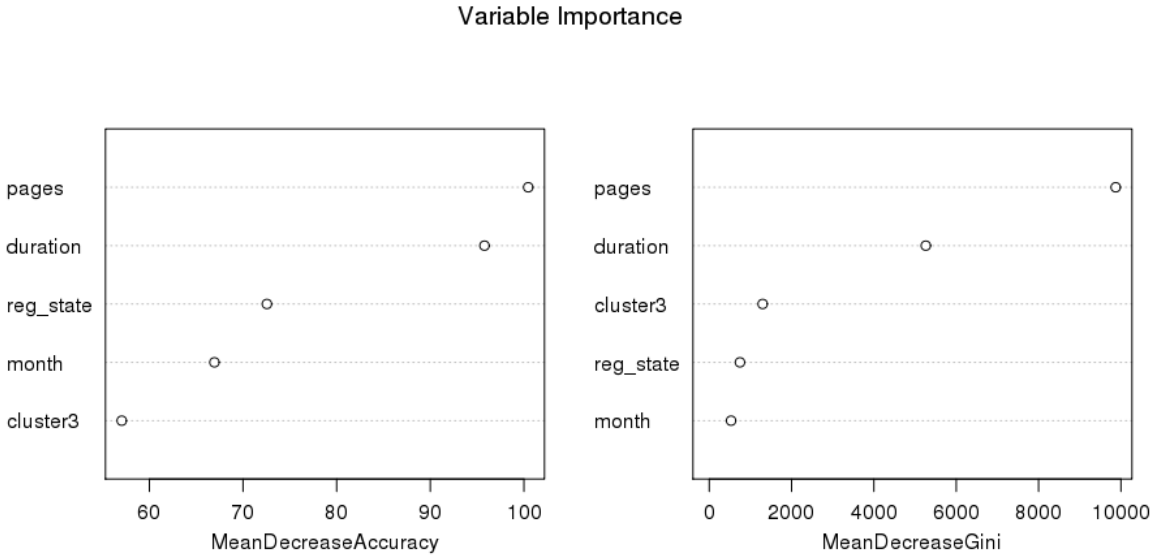


Figure 14: Variable Importance in Random Forest

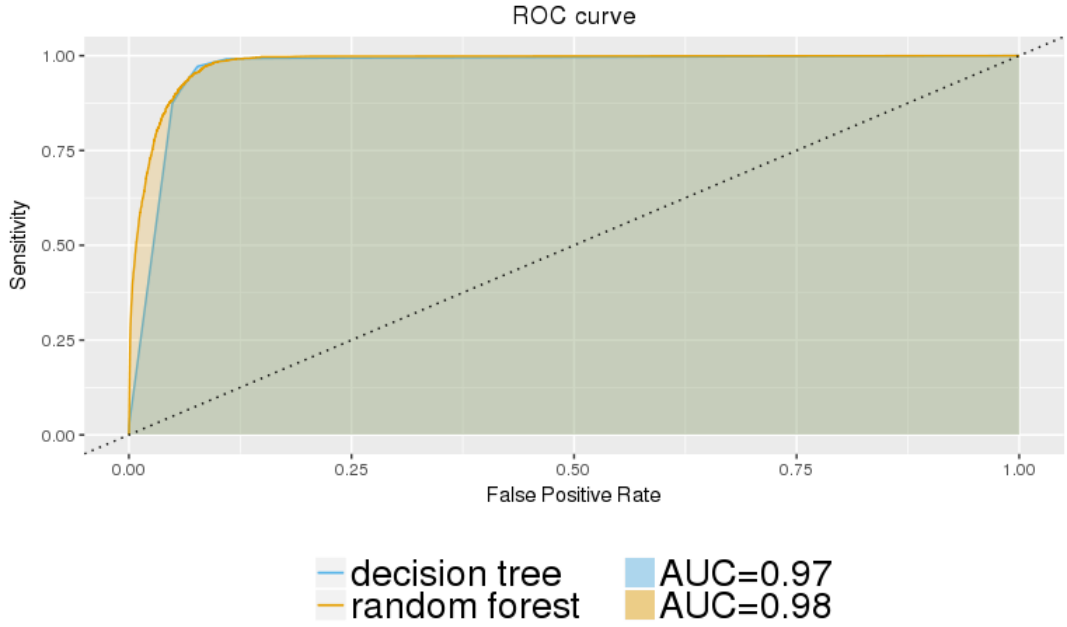


Chapter 6

Conclusion and Remarks

In this article, we have analyzed $667,068$ unique visitors' browsing behaviors on the website of a wine retailer and derived two modern machine learning model, decision tree and random forest, to predict their purchase decision. After re-sampling, both models reach to high predictive accuracy with 92.35% and 93.69% respectively. Random Forest intuitively has higher predictive power than Decision Tree due to its stochastic and collective property. On the other hand, although the sensitivity of decision tree is slightly higher than random forest, we are inclined to accept the random forest model since it has higher F1 score and AUC under ROC curve, which is shown comparatively in figure 15, and naturally evades the over-fitting problem. We also show that the number of viewed pages and the registered state are the most critical feature which will determine whether a visitor will send an order or not. Although the feature of sequence clustering does not significantly influence the accuracy of the model, it still provides a new insight of reviewing visitors' browsing behaviors on the website. Moreover, we successfully construct all the classifiers via the open source packages in R, which also justify the growing popularity of R in data science. R grants the easy code sharing and reusing which improves the reproducibility of research. The package we utilize, caret, is one of the most popular packages of predictive models since it provides the standard interface for many prediction models.

Figure 15: ROC curve from both models



There are some limitations needed to be concerned. First, the prediction model in machine learning is unlike the regression model which can interpret the marginal effect of features, so we are not able to conclude the statement such as increasing 1 unit of a viewed page will lead to increase a specific probability of sending an order. However, thinking in another way, this is one of the advantages of these two models that we do not need any assumption of distribution on parameters before implementing them. Second, we actually lose a lot of browsing history due to the maximum capacity of a matrix, most information from the clustering result is replaced by the number of viewed pages in the model. The result seems too intuitive as the visitors having many viewed pages are tend to send an order eventually, but the implementation of machine learning is more like selecting a model that forecasts a new observation to the correct classification accurately. If the model is simple to derive an accurate prediction, then it is better in the principle of learning. Nevertheless, if we are able to exceed the matrix capacity of R or find an efficient way to construct the distance matrix, our result can definitely become more informative and convincing.

For the future works, we will extract more features providing the products information

viewed by visitors such as the maximum price or product types viewed by visitors. These kinds of features may contain more detailed information about visitors' incentives to send an order. Moreover, it is interesting to combine with the model that predicts the probability of visitors sending an order instead of merely predicting "Yes" or "No", such as the logistic regression model in machine learning, and it is also necessary for us to apply other alternative classification models, such as Support Vector Machine(SVM) or Neural Network, to verify the optimal prediction model. To extract more detailed features and implement comprehensive classification model will be the main topic on future studies.

References

Andersen, Jesper, Anders Giversen, Allan H Jensen, Rune S Larsen, Torben Bach Pedersen, and Janne Skyt. 2000. "Analyzing clickstreams using subsessions." In *Proceedings of the 3rd ACM international workshop on Data warehousing and OLAP.*: 25–32, ACM.

Bucklin, Randolph E., James M. Lattin, Asim Ansari, Sunil Gupta, David Bell, Eloise Coupey, John D. C. Little, Carl Mela, Alan Montgomery, and Joel Steckel. 2002. "Choice and the Internet: From Clickstream to Research Stream." *Marketing Letters* 13 (3): 245–258. ⇝.

Bucklin, Randolph E., and Catarina Sismeiro. 2009. "Click Here for Internet Insight: Advances in Clickstream Data Analysis in Marketing." *Journal of Interactive Marketing* 23(1): 35 – 48. ⇝, Anniversary Issue.

Chawla, Nitesh V, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. 2002. "SMOTE: synthetic minority over-sampling technique." *Journal of artificial intelligence research* 16: 321–357.

e Costa Magalhães Teixeira, Ricardo Filipe Fernandes. 2015. "Using Clickstream Data to Analyze Online Purchase Intentions." *FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO*.

Drummond, Chris, Robert C Holte et al. 2003. "C4. 5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling." In *Workshop on learning from imbalanced datasets II*. 11, Citeseer Washington DC.

Gabadinho, Alexis, Gilbert Ritschard, Nicolas Séverin Mueller, and Matthias Studer. 2011. "Analyzing and visualizing state sequences in R with TraMineR." *Journal of Statistical Software* 40(4): 1–37.

Hop, Walter, and Michel van de Velden. 2013. "Web-shop order prediction using ma-

chine learning." Ph.D. dissertation.

Kumar, T Vijaya, and HS Guruprasad. 2015. "Clustering of Web Usage Data using Hybrid K-means and PACT Algorithms." *Our Major Indexing at International Level* 4852, p. 871.

Lunardon, Nicola, Giovanna Menardi, and Nicola Torelli. 2014. "ROSE: A Package for Binary Imbalanced Learning.." *R Journal* 6(1).

Menardi, Giovanna, and Nicola Torelli. 2014. "Training and assessing classification rules with imbalanced data." *Data Mining and Knowledge Discovery*: 1–31.

Moe, Wendy W., and Peter S. Fader. 2004. "Dynamic Conversion Behavior at E-Commerce Sites." *Management Science* 50(3): 326–335. ⇝.

Rizwan, Tanzim. 2017. "Purchase Predicting with Click Stream Data." Ph.D. dissertation, East West University.

Sismeiro, Catarina, and Randolph E. Bucklin. 2004. "Modeling Purchase Behavior at an E-Commerce Web Site: A Task-Completion Approach." *Journal of Marketing Research* 41(3): 306–323. ⇝.

Verheijden, Ruud. 2012. "Predicting purchasing behavior throughout the clickstream." *Eindhoven University of Technology, Identity*(0609445), p. 32.

Appendix A

Result of Decision Tree

Table 8: Decision Tree with Over-Sampling Method

N=134,551	**Predicted: No**	**Predicted: Yes**
Actual: No	122,945	10,260
Actual: Yes	38	1,308

Table 9: Decision Tree with Under-Sampling Method

N=134,551	**Predicted: No**	**Predicted: Yes**
Actual: No	122,945	10,260
Actual: Yes	38	1,308

Table 10: Decision Tree with ROSE-Sampling Method

N=134,551	**Predicted: No**	**Predicted: Yes**
Actual: No	126,804	6,401
Actual: Yes	215	1,131

Appendix B

Result of Random Forest

Table 11: Random Forest with Over-Sampling Method

N=134,551	**Predicted: No**	**Predicted: Yes**
Actual: No	132,571	634
Actual: Yes	864	482

Table 12: Random Froest with Under-Sampling Method

N=134,551	**Predicted: No**	**Predicted: Yes**
Actual: No	122,223	10,982
Actual: Yes	26	1,320

Table 13: Random Forest with ROSE-Sampling Method

N=134,551	**Predicted: No**	**Predicted: Yes**
Actual: No	121,329	11,876
Actual: Yes	56	1,290