Predicting Customer Class using Customer Lifetime Value with Random Forest Algorithm

Than Than Win[1], Khin Sundee Bo [2]
University of Information Technology, Myanmar
Email: thanthanwin@uit.edu.mm[1], sundeebo@uit.edu.mm[2]

Abstract

As there are a lot of booming online retailers in e-commerce industry in the Internet age, the need of maintaining competitive advantages has become to pay attention to customer relationship management (CRM). To build a successful CRM strategy, it is needed to know individual customer class which can be calculated from Customer Lifetime Value (CLV): the monetary value of customers purchased from the business during their lifetime. CLV modelling allows us to identify customer's predicted business value. It provides the retailers for effectively allocating the resource in their business. This predictive model has been taken on the global Super Store Retail dataset with almost ten thousand transactions. Our model will predict the customers' class of the next year based on their CLV that will help the online retailer to decide which customer should be invested to get long term CRM. Random Forest (RF) algorithm is utilized to train our model and Random Search tuning is conducted to get the best predictive accuracy. The experimental analysis is performed to compare with AdaBoost algorithm on the same dataset.

Key Words- Customer Lifetime Value, Random Forests, AdaBoost

1. Introduction

Today is Internet age, technology become well developed and people like shopping online more than at the physical store because they can do the price comparisons by browsing through dozens of different websites to find the best price, and shopping online is very convenient. This change drives to the large increase of the retailers in the online retail industry. As cost to acquire a new customer is more expensive than retaining an existing customer, online retailers should focus on their existing customers. But there will be some customers whose costs of marketing, selling, and servicing can exceed the profit from them. Therefore, online retailers should focus only on their highest business value customers to maintain long term CRM. The business value of a customer is often expressed with CLV which represents the total amount of money a customer is expected to spend in business during their lifetime. CLV helps us to solve many problems, such as decisions related to segmentation, addressing, retaining, and acquiring customers, or issues concerning a company's long-term value [7].

Most researchers have implemented CLV problems as regression task but predicting categories of CLV (customer class) as classification analysis is more informative in decision making process which can help the retailer to allocate the marketing spend effectively for their business strategies. The CLV of the individual customer can be calculated from the customers' purchase behaviours that can be captured from the historical transaction data [1]. This prediction model has been conducted based on the retail transaction dataset employed with RF Classification algorithm to forecast the business class of customers for the next year. The dataset used to train this predictive model consists of four years (from 2011 to 2014) transaction data from www.kaggle.com.

To build this predictive model, dataset preprocessing is made firstly that includes cleaning the data, setting interval (feature and target period for training and testing), aggregating the data, discretizing (target variable) and feature extraction (predictor variables). The target variables are calculated by discretizing of CLV. And then, merge the datasets of Feature and Target period for training and testing respectively. After that, feature selection is made on the training set to select the most related features which contribute most to our predictand variable. Initially, our model is trained using RF Classifier algorithm with its default hyperparameters. The training target and features are used to learn the parameters of the model. The parameters from the training period are applied to the model for testing features to predict target variable for the testing period. Afterwards, Random Search cross validating is conducted to find the optimal hyperparameter values of Random Forest to achieve the best accuracy. Finally, performance evaluation of our model is made to compare with another ensemble method called AdaBoost algorithm on the test dataset.

We have organised the rest of this paper in the following way: Section 2 will review the work related with the CLV prediction. Section 3 will describe the dataset, data pre-processing and methods used in this paper: the explanation of feature selection method, RF algorithm, Random Search to tune the best hyperparameters of RF algorithm. After that, the performance measure and results of our model are represented in Section 4. Discussion about our model can be found in Section 5.

2. Related work

(Bernat, 2019) compared the predictive power of three different models: Pareto/NBD, Cox proportional hazard, and Gradient tree boosting to predict CLV for individual customers [1]. The models are trained to predict customer spend in the next year using the historical transactions. In addition to RFM variables, they use other covariates. Among them, The Pareto/NBD extension model performed better than others.

(Jasek P, 2018) discussed the predictive abilities of CLV models by the comparison of the performance of different predictive models: Extended Pareto/NBD model, Markov chain model and Status Quo model based on the six datasets [7]. In that paper, the performance of the models was evaluated for both long and short period. The EP/NBD model outperformed other models in a majority of evaluation metrics.

(Chamberlain, 2017) developed the CLV prediction system for UK based global e-commerce company that uses rich features for the past three-year data to predict the net spend of customers of the next year modeling with RF regression algorithm [2]. It addressed their two problems: (CLV and churn prediction) and results were evaluated. And then, they use feature learning to improve their model by experimenting a hybrid model that combines logistic regression with a Deep Neural Network.

(Nicolas Glady, 2008) defined a churner as someone whose CLV is decreasing [9]. Contribution of that paper introduced a new loss function wherein the loss incurred by the CLV decrease will be used to assess the cost to misclassify a customer. They compare the performance of five classifiers: logistic regression, multi-layer perceptron neural network, decision tree, cost-sensitive decision tree and the AdaCost boosting. The AdaCost classifier and the cost-sensitive tree achieved the best results in their empirical application. Neural network and the decision tree give the best results by AUROC.

As we want to generate accurate predictions of individual customer class, we chose Random Forest algorithm because it provides better predictive performance compared to other supervised learning algorithms. Random forest can give the best results even if the dataset has many features. We explored the many datasets and finally found and choose the US super store dataset which has such characteristics contains no of 24 features and 25000 transactions records.

3. Model Implementation

This section contains the explanation of the steps of how our prediction model was developed, which is illustrated in Figure 1.

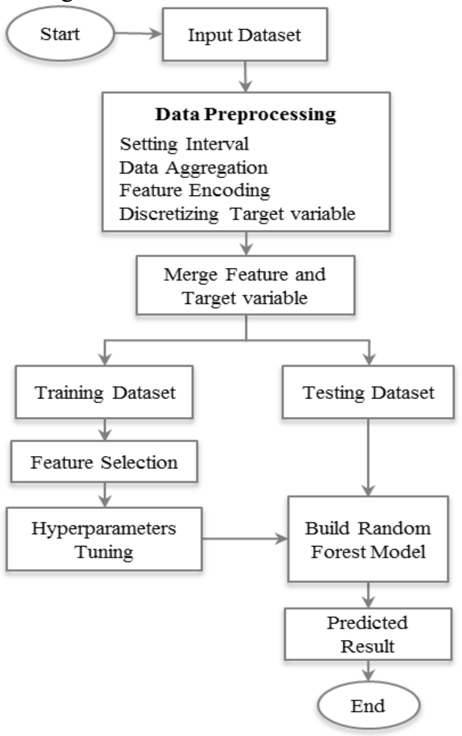

Figure 1 Overview design of the proposed system development

First of all, the description and cleaning of the dataset is given. Next, the data are prepared to be a suitable format for using in the models. And then, the target variable is discretized to identify more informative customer class. After that, applying RF classifier algorithm, conducting hyperparameter tuning and a brief overview of the development environment are described.

3.1. Description and cleaning of the dataset

The global Superstore dataset from Kaggle is adopted to compute the individual CLV of the customers. It consists of 51300 order_line rows wherein totally 25000 unique Order transactions that were made by 1500 customers within a purchase period of 4 years from 1 Jan 2011 to 12 Dec 2014.
The original dataset contains the 24 attributes: Row ID, Order ID, Order Date, Ship Date, Ship Mode, Customer ID, Customer Name, Segment, City, State, Country, Postal Code, Market, Region, Product ID, Category, Sub-Category, Product Name, Sales, Quantity, Discount, Profit, Shipping Cost, Order Priority. In order to get a set of customer behavioral features for our model, we drop the order related attributes such as Order Priority, Category, Product Name, etc. that cannot improve our model performance. If a categorical attribute has too many unique values, it can make tree-based algorithms' predictive power diminished. We explored the unique values of each categorical attributes and drop high unique values attributes and are described in Table 1.

Table 1 Count of unique values of Categorical attributes

Categorical Attributes	Unique Values
Segment	3
Country	141
City	2592
State	41
Region	13

We dropped 'Country', 'City, 'State' and 'Region'. In order to get more insight of customers' purchase behavior, Figure 2 and Figure 3 show that that the number of orders increasing slightly every year and number of customers per frequency by customers that give more understanding of customer purchase behaviour.

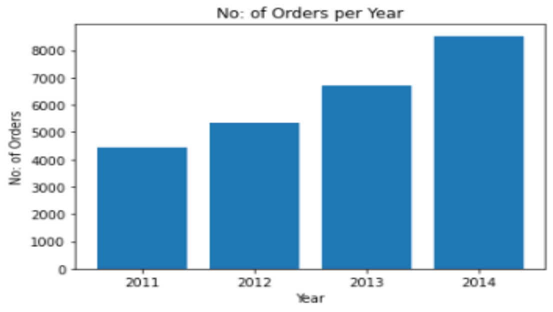

Figure 2 Number of orders per year

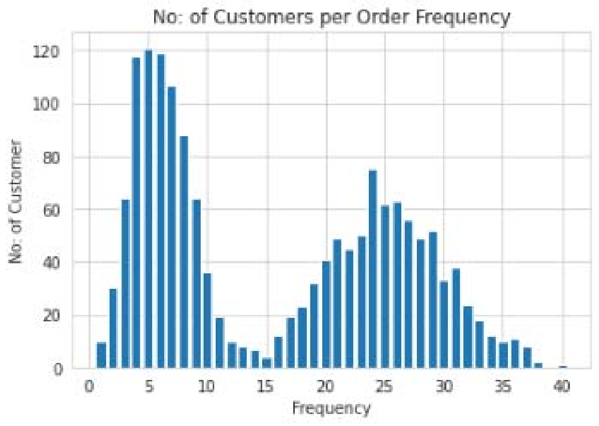

Figure 3 Number of customers per frequency

3.2. Data pre-processing

Initially, Order date of the dataset is changed into datetime format because we need to do some mathematical operation to calculate recent number of purchased days of each customer. We set the Feature and Target period for training and testing: (2011-01-01 to 2012-12-31), (2013-01-01 to 2013-12-31), (2012-01-01 to 2013-12-31) and (2014-01-01 to 2014-12-31) are created as new datasets for each period which is shown in Figure 4. For all the four period datasets, we create 'Orders' dataframe, which consists of all unique orders from the original dataset grouping Order_lines by Order_Id, the variables Total_Amount is created by the sum of Sales (Product's selling price) Column, sum of Profit as Total_Profit and sum of Discount as Total_Discount. Using Orders dataset, we create new dataframe 'Customers' by grouping by Customer_ID in order to get unique customers.

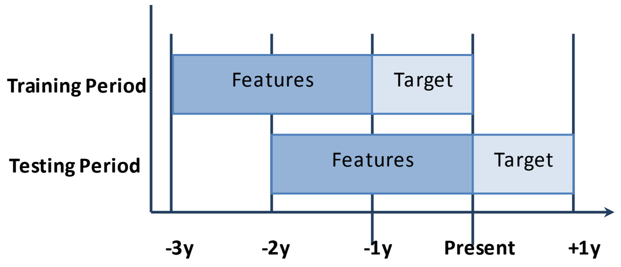

Figure 4 Training and testing period of our model

And then, calculate their Recency (total number of days which is measured as the day of a customer's last purchase minus the last day of observation period), Frequency (the number of orders a customer made during the observation period) and Monetary value (the sum of Total_Amount of all orders made by a customer). The dependent variable 'CLV' is calculated based on the Customers dataset using the following equations (1) to (7).

CLV = (Customer Value/Churn Rate)*Profit margin (1)

Customer Value = Average Order Value * Purchase Frequency (2)

Churn Rate =1 – Repeat Rate (3)

Profit margin = Total Revenue* profit Rate (%) (4)

Average Order Value = Total Revenue/No: of Transactions of individual customer (5)

Purchase Frequency = Total No: of Transactions of all customer/No: of Customers (6)

Repeat Rate = The rate of customers their transaction count are greater than one (7)

Resulted continuous CLV values are discretized to classify the customer class as '0 - (High Class customer)' or '1 - (Low Class customer)' that is used as the label for our model. That column will describe which customer class should be invested for the retention strategy in the upcoming years. The three rows randomly selected from the preprocessed data are described in Table 2.

Table 2 Three rows with discretization of target variable

Customer ID	AB/10015	AB/10105	AB/10255
Revenue	5922.034	5832.223	3845.484
Profit	279.4568	1277.752	-885.905
Discount	4.52	6.3	2.64
Shipping Cost	989.79	975.65	421.02
Frequency	14	14	13
Segment	Consumer	Consumer	Home Office
Recency	3	13	2
Customer Class	0	0	1

According to some categorical variables in our feature set, we conducted one-hot encoding to changes categorical data to a numerical format in order to understand for our model. It spreads the values in a column to multiple flag columns and assigns 0 or 1 to them. These binary values express the relationship between grouped and encoded column.

The last step of pre-processing is feature selection: choosing the most relevant features which contribute most to our predictand variable. We further applied feature importance to rank the most important variables from our features set using the Random Forest feature importance method, which ranks the contribution level of each feature on the prediction of predictand variable '0' or '1'. Six features out of total features are selected and their importance values are mapped in Table 3. We used these features to train our model.

Table 3 Feature importance of selected features

Selected Features	Importance
Revenue	0.203783
Shipping Cost	0.180218
Frequency	0.164207
Profit	0.144875
Recency	0.137844
Discount	0.124139

3.3. Random forest

Random forest is a supervised machine learning algorithm which can be used for both regression and classification problems. It is an ensemble learning methods that combines multiple decision trees, trains each one on a different set of the samples which are drawn with replacement, splitting nodes in each tree considering a limited number of the features, gets a predicted outcome from each decision tree and calculates the votes for predicted outcome. The maximum voting (for classification) or an average (for regression) of the predictions of each individual tree gives the more accurate final prediction.

In Random Forest algorithm, there are two stages: (1) random forest creation, (2) making a prediction from the random forest classifier created in the first stage.

Random Forest creation pseudocode:

- First randomly selects 'k' features from all features
- Among the "k" features, calculate the node "d" using the best split point
- Split the node into child nodes using the best split
- Repeat the upper three steps until "l" number of nodes has been reached

- Build forest by repeating the upper steps for "n" number times to create "n" number of trees

The random forest prediction pseudocode:

- Take the test features and use the rules of each randomly created decision tree to predict the outcome and store the predicted outcome (target)
- Calculate the votes for each predicted target
- Consider the high voted predicted target as the final prediction from the random forest algorithm

3.4. Hyperparameter tuning

For machine learning models, optimizing hyperparameters is a key step to acquire the accurate results. In contrast, model's parameters are values estimated during the training process that parameters specify how to transform the input data into the desired output and hyperparameters define structure of the model that can impact model accuracy and computational efficiency. The models can have many hyperparameters and finding the best combination set of parameters is called hyperparameter tuning.

The hyperparameters of random forest are max_samples (Number of samples train each decision tree), max_features (The number of features to consider when looking for the best split), n_estimators (The number of trees in the forest), criterion (The function to measure the quality of a split), max_depth (The maximum depth of the tree), min_samples_leaf (the minimum number of samples required to be at a leaf node) and min_samples_split (the minimum number of samples required to split). We used Random Search to explore the best hyperparameter sets for our model because of their advantages of improved exploratory power, finding the optimal value for the critical hyperparameter and much lesser time. Random search is a technique where random combinations of the hyperparameters are used to find the best hyperparameter set for the model. It yields better results to Grid Search tuning comparatively. The initialization and the optimal set of hyperparameters are described in Table 4. After the best hyperparameter values are found, the model is trained once again with the training set. Once the model is trained, the testing set is inputted into the model to obtain the loyalty class of each customer.

3.5. Development environment

Python programming language with Spyder Integrated Development Environment was used to implement our predictive model. The Scikit-learn library which provides a range of supervised and unsupervised learning algorithms is utilized to conduct the necessary function in our development.

Table 4 Initialized values and optimal hyperparameters values of random search of model

Hyperparameter	**Initialization**	**Best Parameter**
criterion	'entropy','gini'	'gini'
max_depth	10-110	50
max_features	'auto', 'sqrt'	'sqrt'
min_samples_leaf	1, 2, 4	2
min_samples_split	5, 10, 20	20
n_estimators	200-2000	200
bootstrap	True, False	True

4. Performance measure and results

Our model predicts the class of customers for whether which customer will be high or low class in order to decide which customer class should give how much offer in the upcoming promotion and marketing campaigns. Effectively allocating marketing spend to each customer class can reduce much cost rather than offering all customers equally. The performance of our model is evaluated based on precision, recall and accuracy. Initially we trained the model with default parameters and then accessed the predictive accuracy of our model by training with optimal hyperparameter sets tuned with Random Search on the test dataset using the classification-report which a key metrics to measure the quality of predictions of classification algorithms, which reports the scores of precision, recall, f1-score and accuracy. Precision describes what proportion of **predicted Positives** is truly positive, Recall expresses what proportion of **actual Positives** is correctly classified, f1-score describes what percentage of positive predictions are correct, Accuracy means what proportion of all Positive and Negative were correctly classified, and their equations are as follows:

$$\textbf{Precision} = \textbf{TP}/(\textbf{TP}+\textbf{FP}) \tag{8}$$

$$\textbf{Recall} = \textbf{TP}/(\textbf{TP}+\textbf{FN}) \tag{9}$$

$$\textbf{f1-score} = 2*(\textbf{Recall}*\textbf{Precision})/(\textbf{Recall}+\textbf{Precision}) \tag{10}$$

$$\textbf{Accuracy} = (\textbf{TP}+\textbf{TN})/(\textbf{TP}+\textbf{TN}+\textbf{FP}+\textbf{FN}) \tag{11}$$

Where, TP is when a Positive sample is correctly classified as Positive class, FP means when a

Negative sample is falsely classified as Positive class, TN is when a Negative sample is correctly classified as Negative class, FN is called when a Positive sample is incorrectly predicted as Negative class.Table 5 describes the accuracy of classifier models with different hyperparameters' sets on testing dataset. In Table 6, we show that comparison result of Random Forest best model and AdaBoost model with respect to precision, recall, and F1.

Table 5 Accuracy of classifier models with different hyperparameters' sets on testing dataset

Random Forest Models	Accuracy
Default hyperparameters	81.46%
Selected feature using Feature Selection with default hyperparameters	82.26%
Random Search's best hyperparameters set	84.27%
AdaBoost Model	78.21%

Table 6 Resulted precision, recall, and f1-score for each customer class of best model

	Precision	Recall	F1-score
0 (High Class)	93.65%	79.45%	85.96%
1 (Low Class)	74.34%	91.70%	82.12%

5. Discussion and conclusion

From Table 5, we can see that the accuracy of model with Random Search's optimal hyperparameter value outperform than AdaBoost models. Model with default hyperparameters values with all features is 81.46%. With only the selected features from feature selection, the model's accuracy improved to 82.26%, that is really good enough model. Random forest model with the optimal hyperparameters tuned by Random Search increased by 2%.

For High class customer, our model performed very well with high precision and good f1-score, and high recall for Low class customers.

In this paper, we provide customer class prediction model using random forest algorithm to correctly classify individual online retail customer class, viewed from the perspective of customer lifetime value. We showed that our model performed consistently well using selected features and best parameters from Random Search. Using our model can help the online retailers to decide which class of customer need to put much effort to maintain retention strategy. As the further study based on the customer's class and exploring their preferences, product interest will be recommended, which leads to increase in sale and helps to develop a better relationship with your potential customers and can incentivize Low class customer to improve our retention strategy.

6. References

[1] J.R. Bernat, A.J. Koning, and D. Fok, "Modelling customer lifetime value in a continuous, non-contractual Time Setting", Netherlands, 2019.

[2] B.P. Chamberlain, A. Cardoso, C.H. Bryan Liu, R. Pagliari, and M.P. Deisenroth, "Customer lifetime value prediction using embeddings," *the 23rd ACM SIGKDD International Conference,* Volume: 23, Canada, 2017.

[3] S. Chen, "Estimating customer lifetime value using machine learning techniques*",* London, 2018.

[4] DATA SCIENCE GROUP, AMPERITY INC, "Predicting Customer Lifetime Value with Unified Customer Data", 2019.

[5] E. Farzanfar and N. Delafrooz, "Determining the Customer Lifetime Value based on the Benefit Clustering in the Insurance Industry". *Indian Journal of Science and Technology*, India, 2016.

[6] C. Jangid, T. Kothari, J. Spear, and E. Wadsworth, "Custoval: estimating customer lifetime value using machine learning techniques," Dept. *of CIS-Senior Design 2013,* 2014.

[7] P. Jasek, L. Vrana, L. Sperkova, Z. Smutny, and M. Kobulsky, "Modeling and application of customer lifetime value in online retail," *Informatics 5,* India, 2019.

[8] M. Karlssson, "Predicting customer lifetime value using machine learning algorithms", 2016.

[9] N. Glady, B. Baesens, and C. Croux, "Modeling churn using customer lifetime value," *KU Leuven KBI Working Paper,* Belgium, 2008.

[10] T. Rathi, "Customer lifetime value measurement using machine learning techniques*",* IGI Global, USA, 2011.

[11] A. Vanderveld, A. Pandey, A. Han, and R. Parekh, "An engagement-based customer lifetime value system for e-commerce," *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining,* New York, 2016.