ELEN E6895 Final Report
Advanced KYC-Customer Behavior Prediction

Jingchao Hu(jh4312), Peihan Liu(pl2804)
Columbia University

Abstract

This article provides a model to analyze customers' behavior. By analyzing the data about customers, including transaction amounts, date, etc, we can divide customers into 3 levels and provide them with appropriate discounts. In this way, we are more likely to get attraction from frequent customers and potential customers.

1. Introduction

1.1. Background

Nowadays, online groceries shopping is becoming more and more popular. Especially under the impact of the Covid-19, many people prefer buying groceries online other than on site, which not only increases the importance of the customer behavior prediction, but also provides us with availability to a large amount of transaction data on online groceries shopping. By utilizing this data, we have the opportunity to model the data and analyze customers' behavior, which is one of the commonest marketing strategies. Apart from that, the thriving artificial intelligence techniques can assist us to provide more reliable and practical marketing strategic plans.

1.2. Problem Statement

Customer behavior prediction is an important topic of marketing. Many companies and research groups have put huge effort into it. In our project, we will utilize a reliable dataset and model the data. By doing so, we can analyze the customer behavior and get a better knowledge about the preferences of each customer. With the result above, we will be able to give customer recommendations on items, customer promotions and know whether the customer is losing interest in our product. We aim to find a suitable algorithm to help us predict customer behavior in the future.

1.3. Dataset

In our project, we will use the Ta Feng Groceries dataset, which contains Chinese grocery store transaction datas from November 2000 to February 2001. This dataset contains 119578 shopping baskets splitting into about 119578 shopping baskets. All these records belong to 32266 users and 23812 products. The dataset also provides other useful information like the age of the customer. We believe this dataset is perfect for us to achieve the goal.

Fields of the Dataset are:

- Transaction date and time
- Customer ID
- Age: 10 possible values
- Residence Area: 8 possible values
- Product subclass
- Product ID
- Amount
- Asset
- Sales price

2. Exploratory Data Analysis

After getting the data from the Ta Feng Groceries dataset, we first conduct an exploratory data analysis so that we can know better about the data pattern and pave the path for the later features engineering.

Week: We come to see the count of the transactions by week to confirm the data is correct and possibly identify the time periods that have the most activity between the two years.There is a great deal of activity in the weeks leading up to the holiday season and then tapering off after several weeks after the holiday season. I would expect the last few weeks of the dataset to reflect more normal purchasing activity compared the rest of the dataset. That being said after doing a test comparing the sample means of purchasing activity for both years, using a threshold of p=0.05, we fail to reject the null hypothesis that there is no difference in the sample means. Therefore we can conclude that the average transaction counts are similar from November/December 2000 and January/February 2001.

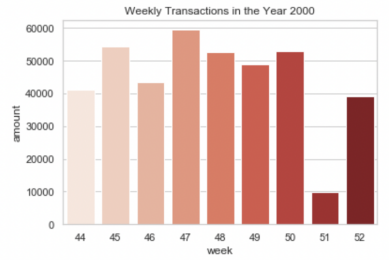

Figure 1. Weekly Transactions in the Year 2000

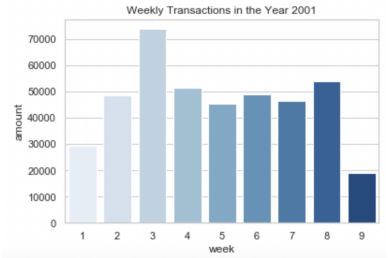

Figure 2. Weekly Transactions in the Year 2001

Age: Then, in the age ranges for customers, we can see that the bulk of frequent shoppers from the age ranges of 35 to 44. This could seem consistent with the fact. For example, a young adult shopper needs to buy cargos both for themselves and their children. While some older adults only need to buy for themselves and partners. And in the regions for customers, we can see the highest amount of transactions happen within 5 and 6.

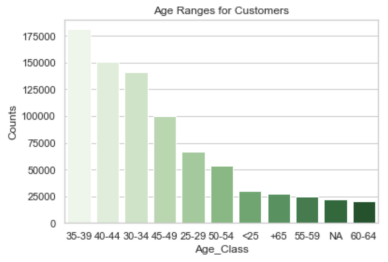

Figure 3. Age Ranges for Customers

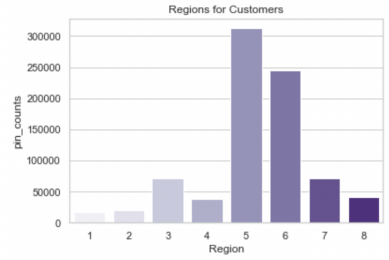

Figure 4. Regions for Customers

Amounts and Sales Prices: After looking at the differences in transaction counts by year. We have come to test the hypothesis that perhaps the means of each year could be different from each other. Then we further look at the distribution of 'Amount' and 'Sales Price' and find there are dramatic spikes in activity for both purchasing 'Amounts' and 'Sales Prices'.

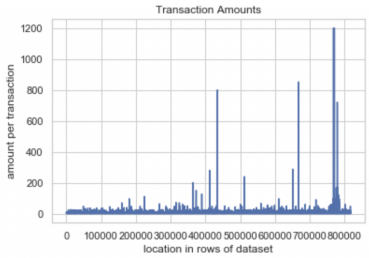

Figure 5. Amounts of Each Transactions

Figure 6. Prices of Each Transactions

Unit Price: After further exploring these behaviors I found that there appeared to be a non-linear relationship with the two variables and decided perhaps 'Sales Price' reflects the total transaction cost and not the individual 'Unit Price' that we previously believed.

Figure 7. Amount Purchased by Sales Price

3. Metrics

Behavioral customer segmentation is based on three metrics.

Recency (R): How many days since customer's last purchase (the lower the better) until the present

Frequency (F): How many purchases the customer has done since their start of the time period

Monetary Value (M): Measures how much the customer has spent since the start of the time period

RFM variables are useful for beginning to classify users based on their behavior over time. Then we calculate the RFM score by adding up the quantile numbers of each R,F,M value(i.e. if Recency is 4th quantile, Frequency is 3rd quantile, and Monetary is 2nd quantile, then the RFM Score would be 9) and classify the customer into 3 levels, each level represents the value of the customer and level 1 and 2 customers are our target customers.

- Level 1 = RFM Score equal to or greater than 9
- Level 2 = RFM Score between 5 and 9
- Level 3 = RFM Score lower than 5

	Recency	Frequency	Monetary	
	mean	mean	mean	count
RFM_Score				
3.0	91.2	3.0	291.9	2742
4.0	63.2	4.5	522.1	2873
5.0	56.1	6.9	890.4	3443
6.0	40.0	8.7	1115.7	3801
7.0	40.3	13.6	1781.9	3458
8.0	30.1	18.0	2420.4	3488
9.0	23.8	24.6	3207.8	3232
10.0	19.3	36.6	4854.4	3135
11.0	12.3	56.1	7150.8	2939
12.0	4.2	85.8	10622.7	3151

Figure 8. Summary Metrics per RFM Score

Now we can see an easier to read table summarized by the relative RFM score. Its much clearer to see how those with a higher RFM score are more important customers to the business.

We use RFM score to group customers into Gold,Silver,and Bronze segments for easier interpretation.

customer_id	Recency	Frequency	Monetary	R	F	M	RFM_Segment	RFM_Score	General_Segment
00001069	19	11	1944.0	3	2	3	323	8.0	Silver
00001113	54	18	2230.0	2	3	3	233	8.0	Silver
00001250	19	14	1583.0	3	2	2	322	7.0	Silver
00001359	87	3	364.0	1	1	1	111	3.0	Bronze
00001823	36	14	2607.0	2	2	3	223	7.0	Silver
00002189	57	62	14056.0	2	4	4	244	10.0	Gold
00003667	21	13	11509.0	3	2	4	324	9.0	Gold
00004282	47	9	967.0	2	2	2	222	6.0	Silver
00004381	103	11	701.0	1	2	1	121	4.0	Bronze
00004947	81	36	3363.0	1	4	3	143	8.0	Silver

Figure 9. Sample of Customer Classification

4. Data Processing

After the EDA process, we will still need to recalculate the RMF values. We set up a series of time windows and the RMF values will be changing at different time windows.

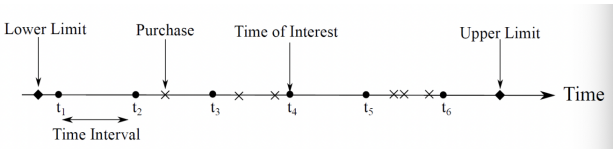

Figure 10. A sample of shopper's behavior during different time intervals

Each transaction has a changing Recency, Frequency, and Monetary value associated from the time since the customer's first purchase. In the figure below at the time of t3 the frequency would be F=1 (I add 1 for each customer's first transaction) and at the time of interest (t4) the frequency would be F=3 rather than a total sum frequency of f=7.

After recording the R, M, F values, our next step is to limit the dataset to matched pairs for training and prediction. Essentially, we want to make sure that every customer ID and RMF value set we train is also represented in the prediction set, and vice versa. We do this because both the customer ID and the RMF value will be used to inform our prediction of the next timestep of the RMF value. We also split the data in time so that the customer ID and RMF values from the previous 15 weeks will be used to predict the next value represented by week 16 or 17. Also, we require that the data must have at least more than the frequency values to be kept in either dataset. Our reasoning is that we want to observe

"regular" shoppers in the dataset, that means at least more than one purchase.

Once we complete this process, we are left with 6669 observations for the independent training variables and 6669 dependent predictors. We then split the customer ID into integers based on the assumption made in the paper that the customer ID is actually a customer loyalty number and may provide valuable information in predicting RMF values.

5. Modeling

We uses recurrent neural networks(RNN) to solve the problem. The reason for that is recurrent neural networks are particularly good at capturing non-linear sequences and have an internal state capable of retaining information from previously learned data structures making them particularly good at data learning structures with a temporal element such as speech recognition, time series prediction, and robot control. Essentially, predicting R,F,M is the exact kind of application that Recurrent Neural Networks should be good at since the R,F,M values certainly have a temporal element since Recency and Frequency are capturing customer activity through time and as you will see the R,M,F values certainly follow a nonlinear sequence.

5.1. Full Model

For the baseline model We chooses the simplest method, which just pass all the features that we feel could help to possibly inform the prediction along with split values of Customer ID and R, M, F values.These values are passed into a function below that splits the Customer IDs into separate integer values, just in case there truly are features within the Customer ID that can inform RFM prediction.

Layer (type)	Output Shape	Param #
simple_rnn_19 (SimpleRNN)	(None, 19)	741
dense_37 (Dense)	(None, 250)	5000
dense_38 (Dense)	(None, 11)	2761

Total params: 8,502
Trainable params: 8,502
Non-trainable params: 0

Figure 11. Full Model Structure

The next step was to turn the data into array format and normalize the data since the RMF values and features were all on different scales. Then I split the data into training and testing sets in order to pass them into the model. As mentioned previously the model type was a Simple Recurrent Neural Network with 250 hidden units with Relu activation using L1 regularization and a loss of Means Squared Error.

The hyperparameter options are very similar to those mentioned in the "Customer Shopping Pattern Prediction" paper. We think the only difference is that we are using the very popular Adam optimizer instead of SGD. Since Adam is based on SGD, we consider this a narrow distinction. We get better results with Adam.

- SimpleRNN
- Relu activation
- 250 hidden units
- L1 regularization at 0.0001
- MSE loss
- Batch size 120
- Shuffle=True
- 1000 epochs

```
5335/5335 [==============================] - 0s 79us/step - loss: 0.0039 - mean_absolute_error: 0.0209 - acc: 0.760
4 - val_loss: 0.0045 - val_mean_absolute_error: 0.0234 - val_acc: 0.7751
Epoch 991/1000
5335/5335 [==============================] - 0s 77us/step - loss: 0.0040 - mean_absolute_error: 0.0215 - acc: 0.757
5 - val_loss: 0.0044 - val_mean_absolute_error: 0.0230 - val_acc: 0.7901
Epoch 992/1000
5335/5335 [==============================] - 0s 76us/step - loss: 0.0040 - mean_absolute_error: 0.0208 - acc: 0.757
3 - val_loss: 0.0045 - val_mean_absolute_error: 0.0241 - val_acc: 0.7736
Epoch 993/1000
5335/5335 [==============================] - 0s 76us/step - loss: 0.0039 - mean_absolute_error: 0.0208 - acc: 0.759
5 - val_loss: 0.0046 - val_mean_absolute_error: 0.0260 - val_acc: 0.7579
Epoch 994/1000
5335/5335 [==============================] - 0s 82us/step - loss: 0.0040 - mean_absolute_error: 0.0230 - acc: 0.758
4 - val_loss: 0.0044 - val_mean_absolute_error: 0.0235 - val_acc: 0.7676
Epoch 995/1000
5335/5335 [==============================] - 0s 84us/step - loss: 0.0040 - mean_absolute_error: 0.0210 - acc: 0.761
0 - val_loss: 0.0044 - val_mean_absolute_error: 0.0239 - val_acc: 0.7781
Epoch 996/1000
5335/5335 [==============================] - 0s 77us/step - loss: 0.0040 - mean_absolute_error: 0.0207 - acc: 0.751
1 - val_loss: 0.0044 - val_mean_absolute_error: 0.0232 - val_acc: 0.7151
Epoch 997/1000
5335/5335 [==============================] - 0s 81us/step - loss: 0.0040 - mean_absolute_error: 0.0234 - acc: 0.755
6 - val_loss: 0.0045 - val_mean_absolute_error: 0.0229 - val_acc: 0.7421
Epoch 998/1000
5335/5335 [==============================] - 1s 102us/step - loss: 0.0040 - mean_absolute_error: 0.0213 - acc: 0.75
78 - val_loss: 0.0044 - val_mean_absolute_error: 0.0205 - val_acc: 0.7129
Epoch 999/1000
5335/5335 [==============================] - 0s 85us/step - loss: 0.0040 - mean_absolute_error: 0.0215 - acc: 0.752
6 - val_loss: 0.0044 - val_mean_absolute_error: 0.0241 - val_acc: 0.7856
Epoch 1000/1000
5335/5335 [==============================] - 0s 92us/step - loss: 0.0039 - mean_absolute_error: 0.0211 - acc: 0.752
6 - val_loss: 0.0045 - val_mean_absolute_error: 0.0233 - val_acc: 0.7714
```

Figure 12. The training process of full model

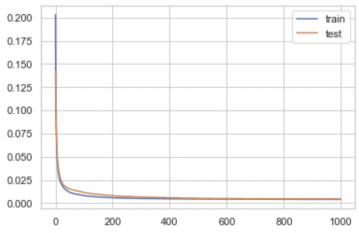

Figure 13. The Learning Curve of Full Model

The results show an overall accuracy of 74%, with a validation accuracy of 76%, and a Root Means Squared Error of 270; which are surprisingly good results considering how little modification I had done to the data.

5.2. Reduced Model

Next I decided to try a 'Reduced Model' with only the bare minimum features; simply the Customer ID and RMF values.

Layer (type)	Output Shape	Param #
simple_rnn_20 (SimpleRNN)	(None, 11)	253
dense_39 (Dense)	(None, 250)	3000
dense_40 (Dense)	(None, 11)	2761

Total params: 6,014
Trainable params: 6,014
Non-trainable params: 0

Figure 14. Reduced Model Structure

```
5335/5335 [==============================] - 0s 83us/step - loss: 0.0048 - mean_absolute_error: 0.0245 - acc: 0.749
6 - val_loss: 0.0098 - val_mean_absolute_error: 0.0479 - val_acc: 0.7211
Epoch 991/1000
5335/5335 [==============================] - 0s 84us/step - loss: 0.0048 - mean_absolute_error: 0.0236 - acc: 0.741
3 - val_loss: 0.0100 - val_mean_absolute_error: 0.0491 - val_acc: 0.7264
Epoch 992/1000
5335/5335 [==============================] - 0s 81us/step - loss: 0.0048 - mean_absolute_error: 0.0240 - acc: 0.747
1 - val_loss: 0.0100 - val_mean_absolute_error: 0.0502 - val_acc: 0.6942
Epoch 993/1000
5335/5335 [==============================] - 0s 84us/step - loss: 0.0049 - mean_absolute_error: 0.0261 - acc: 0.744
5 - val_loss: 0.0098 - val_mean_absolute_error: 0.0473 - val_acc: 0.6897
Epoch 994/1000
5335/5335 [==============================] - 0s 83us/step - loss: 0.0048 - mean_absolute_error: 0.0234 - acc: 0.743
0 - val_loss: 0.0100 - val_mean_absolute_error: 0.0486 - val_acc: 0.7151
Epoch 995/1000
5335/5335 [==============================] - 0s 80us/step - loss: 0.0048 - mean_absolute_error: 0.0239 - acc: 0.747
1 - val_loss: 0.0099 - val_mean_absolute_error: 0.0500 - val_acc: 0.6829
Epoch 996/1000
5335/5335 [==============================] - 0s 88us/step - loss: 0.0049 - mean_absolute_error: 0.0257 - acc: 0.746
6 - val_loss: 0.0099 - val_mean_absolute_error: 0.0485 - val_acc: 0.7204
Epoch 997/1000
5335/5335 [==============================] - 0s 83us/step - loss: 0.0048 - mean_absolute_error: 0.0238 - acc: 0.745
6 - val_loss: 0.0099 - val_mean_absolute_error: 0.0490 - val_acc: 0.7024
Epoch 998/1000
5335/5335 [==============================] - 0s 84us/step - loss: 0.0048 - mean_absolute_error: 0.0239 - acc: 0.743
0 - val_loss: 0.0100 - val_mean_absolute_error: 0.0495 - val_acc: 0.7264
Epoch 999/1000
5335/5335 [==============================] - 0s 86us/step - loss: 0.0048 - mean_absolute_error: 0.0246 - acc: 0.741
0 - val_loss: 0.0100 - val_mean_absolute_error: 0.0503 - val_acc: 0.6942
Epoch 1000/1000
5335/5335 [==============================] - 0s 82us/step - loss: 0.0048 - mean_absolute_error: 0.0243 - acc: 0.748
8 - val_loss: 0.0098 - val_mean_absolute_error: 0.0472 - val_acc: 0.6949
```

Figure 15. The training process of reduced model

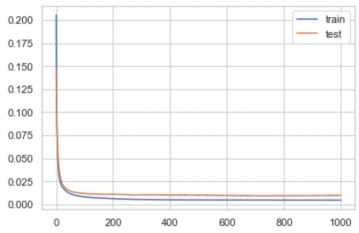

Figure 16. The Learning Curve of Reduced Model

6. Conclusion

After the comparison of full model and reduced model, we can make conclusion:

- Very little of the model is informed by the inclusion of all the additional features(Week_number, Amount, Total_sum, Cluster, Age_group, Pin_code, Unit_price, Log_unit_price); A fact that I find surprising alone. The only benefit to the model by adding the other features is perhaps a reduced Root Means Squared Error for prediction.
- Using very few previous transactions you can predict the R,F,M values at at least a 70% accuracy. Of course, if I were trying to build a model to diagnose cancer, I'd probably throw it away and go back to formula, but since we are probably dealing with advertising and promotions I feel 70% is plenty accurate to decide to provide a 20% discount to regular shoppers at Bath and Bodyworks .
- Recurrent Neural Networks are very good at sequences!

7. Frontend Part

We use our trained reduced model to infer the test set of size of 1334 customer IDs. Based on the prediction, we calculate the RFM score and segment the customers into gold, silver and bronze level. Then we utilize flask to bulit a web application, which gets connections with the front end and back end.In our simple web, the sales will input the id of the customer and total amount that he spent this time. After submitting, it will jump to the page to calculate the cost after the discount. If the level of customer is golden state, he will enjoy a twenty percent off, for example, if he spent 100 dollars this time, he would factly pay 80 dollars. Then as for the silver level's customer, he needed to pay 90 dollars in this case. And others needed to pay full price.

References

Pouladi, F., Salehinejad, H., and Gilani, A. M. Recurrent neural networks for sequential phenotype prediction in genomics. In *2015 International Conference on Developments of E-Systems Engineering (DeSE)*, pp. 225–230. IEEE, 2015.

Salehinejad, H. and Rahnamayan, S. Customer shopping pattern prediction: A recurrent neural network approach. In *2016 IEEE symposium series on computational intelligence (SSCI)*, pp. 1–6. IEEE, 2016.

Zhang, Y., Bradlow, E. T., and Small, D. S. Predicting customer value using clumpiness: From rfm to rfmc. *Marketing Science*, 34(2):195–208, 2015.