Project: Capstone Project 2: Milestone Report 2

Customer Purchasing Behavior

During the course of our lives most of us one point or another have had to buy at least one product from Walmart. However, for the lucky few who haven't heard of Walmart; Walmart is the poster child for multi-national big-box retail store that stock and sell over 75 million different products, items ranging from paint to plushies, from guns to guacamole. With over 11,300 stores globally, employing 2.2 million people, and with 275 million weekly customers, Walmart is easily one of the largest franchises in the world. Walmart's fiscal revenue is something to the tune of $514 billion per year making it the most lucrative company in the world. With nearly every Walmart store selling grocery items and produce, what many may not have considered is that Walmart is also one of the world's largest U.S. grocery retailers.
https://en.wikipedia.org/wiki/Walmart
https://expandedramblings.com/index.php/walmart-statistics/

Problem Statement:

Customer purchasing optimization is a common problem for a lot of companies, both large and small. Luckily most large companies should have a wealth of data available to them in order to predict the possible purchasing behavior for many of their clients so they can make informed decisions about stocking their shelves, providing customer recommendations on items, customer promotions, and predicting when a customer is losing interest in their service and planning to buy from a competitor. My proposal is to analyse and demonstrate some of these metrics on public data found on the internet as well as provide an algorithm that can help to predict future long-term purchasing in the form of Customer Lifetime Values.

Dataset:

Inspired by the paper, Customer Shopping Pattern Prediction: A Recurrent Neural Network Approach by Hojjat Salehinejad and Shahryar Rahnamayan, the authors used a Recurrent Neural Network to predict customer loyalty values (R,F,M) using the **Ta Feng Grocery Dataset**. After doing some internet searching I managed to find a copy or sample of the Ta Feng Grocery dataset hosted on Kaggle but not part of a competition. The Ta Feng Dataset is a Supermarket Dataset containing 817741 transactions from November 2000 until the end of February 2001. The dataset contains information about 119578 shopping baskets, belonging to 32266 users, where 1129939 items were purchased from a range of 23812 products.

Summary:

As previously mentioned this work was inspired by Customer Shopping Pattern Prediction: A Recurrent Neural Network Approach by Hojjat Salehinejad and Shahryar Rahnamayan. Together they managed to put together a short, legible, and impactful paper that inspired me to try predicting R, M, and F values. In my opinion journal articles such as this need to get published more often. Using the Ta Feng Grocery data the authors simply used the Customer ID's and past R, M, F values for their prediction which were assumed as the Customer Loyalty Numbers (CLN); provide enough information to be able to predict the (R) Recency, (F) Frequency, and (M) Monetary values one step forward in time. In their design the authors encoded the Customer IDs as dummies (or one-hot encoding), then passed the dummies along with the R,M,F values into an auto-encoder, shrinking the size of the data, before passing the data into a Simple Recurrent Neural Network for regression. In so doing, the authors achieved an 80% accuracy for their model.

My approach was even simpler if you can believe it. I started with recoding the R, F, and M values to match the style the authors used in the paper and simply passed all the data using a Simple Recurrent Neural Network and similar hyper-parameters. To my surprise I was provided with a 75% accuracy. On one hand I was surprised to find that the accuracy wasn't higher since I certainly included more features than cited in the paper, including even some features with collinearity. I was also surprised at how little work was needed to get those results. After reducing the data to just Customer ID's alone with R,M,F values; to my surprise I again achieved a 75% accuracy. This told me that very little of the model was being informed by the inclusion of these additional features. The only added benefit that I found of adding more features was perhaps a reduced RMSE value.

My Conclusion:

With these findings I feel the added bonus of a reduced RMSE isn't truly more beneficial than simply using the Customer ID's and previous R, M, F values. I also feel more inclined toward the feeling that maintaining a simpler model is just better. I also have a *(perhaps controversial)* feeling that after these results, the added complication of one-hot encoding dummy variables from the Customer ID's and using dimensionality reduction techniques to predict R, M, F values, *simply for a 5% increase in accuracy*, to be lackluster compared to the simplicity of just passing the data into the model and getting a 75% accuracy.

Data Wrangling:

After importing the proper packages I downloaded the dataset and all the label dictionaries I used after my data exploration in order to continue my work. After getting the data imported I set about recoding recalculating the R, M, F values.

If you read my previous milestone report, you may be asking why am I calculating this yet again?

Because here I am doing something slightly different, for each window of time I am calculating the number of days since the customer's first transaction rather than the flat (snapshot) metric that I had calculated in the data exploration notebook.

So what that means is 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**.

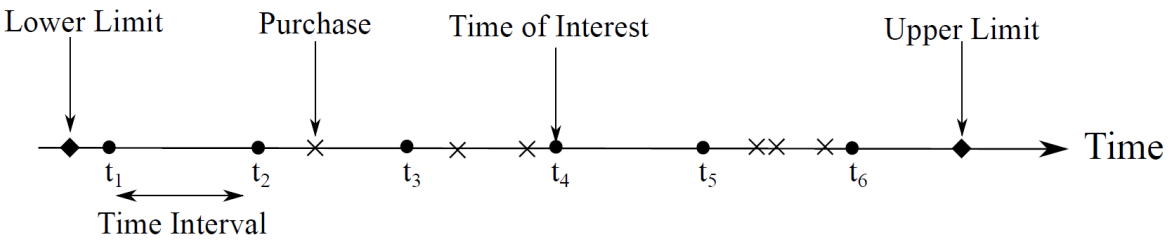


Fig. 3: A sample of shopper's behaviour during different time intervals.

After recoding the R, M, F values my next step was to restrict the dataset to matching pairs for training and prediction. Essentially, I wanted to make sure that each Customer ID and RMF value set that I trained

on was also represented in the prediction set and vice versa. I did this because both the Customer ID and the RMF values will be used to inform my prediction of the next time step of RMF values. I split the data in time as well, so that the Customer IDs and RMF values in the first 15 weeks will be used to predict the next value represented in weeks 16 or 17. Furthermore, I require that the data has to have at least more than one frequency value to be kept in either dataset. My reasoning is that I would like observations of 'regular' shoppers in the dataset; and that means at least more than one purchase.

Once I completed this process I am left with 6669 observations for the independent training variables and 6669 dependent prediction variables. I then split Customer IDs into integers based on the assumption presented in the paper that the Customer IDs are in fact Customer Loyalty Numbers and may present valuable information in predicting RMF values.

Modeling:

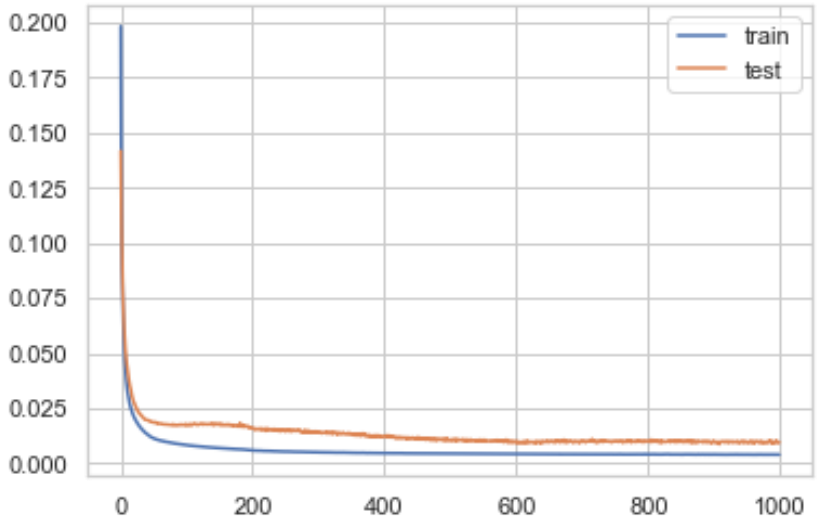
For the first run I chose, in a way, the simplest method; just pass all the features that I feel could help to possibly inform the prediction along with split values of Customer ID and R, M, F values. *(Let's call it the kitchen sink approach)* These values are passed into a function below that splits the Customer IDs into separate integer values, just in case there truly are features within the Customer ID that can inform RFM prediction.

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.

Hyperperameters
1. SimpleRNN
2. Relu activation
3. 250 hidden units
4. L1 regularization at 0.0001
5. MSE loss
6. Batch size 120
7. Shuffle=True
8. 1000 epochs

```
_________________________________________________________________
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
_________________________________________________________________
```

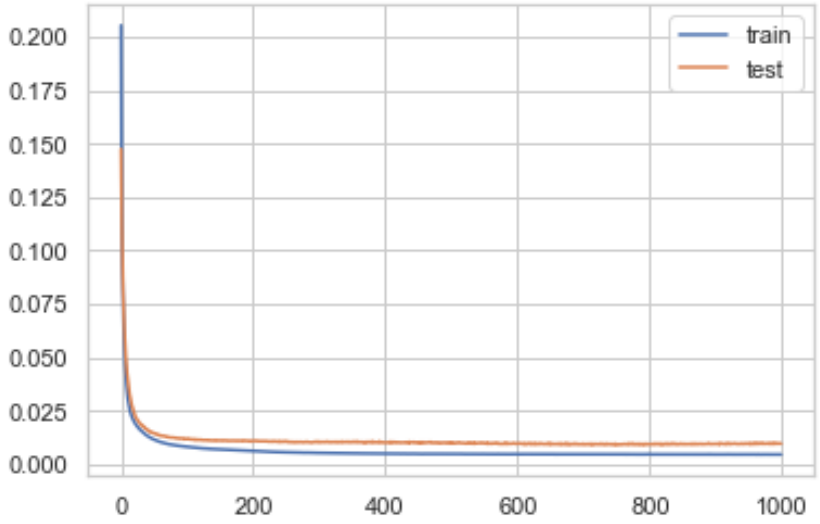
Full Model




```
Epoch 1000/1000
5335/5335 [==============================] - 1s 124us/step - loss: 0.0040 -
mean_absolute_error: 0.0213 - acc: 0.7445 - val_loss: 0.0096 -
val_mean_absolute_error: 0.0460 - val_acc: 0.7676
Test RMSE (Prediction): 270.461
```

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

Reduced Model



Epoch 1000/1000
5335/5335 [==============================] - 0s 82us/step - loss: 0.0048 - mean_absolute_error: 0.0243 - acc: 0.7488 - val_loss: 0.0098 - val_mean_absolute_error: 0.0472 - val_acc: 0.6949

After running both models we see some surprising results! Firstly, I am surprised at how well the Simple RNN model is able to figure out the relationships with sequences with such little information (Reduced Model and Journal Article). From the start I get at least a 70% accuracy without the use any additional features and dimensionality reduction techniques or a wealth of previous transaction information.

I after comparing the Full and Reduced models I think we can make some simple conclusions:

1. 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.
2. 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 (*we all know it to be true*).
3. Recurrent Neural Networks are very good at sequences!