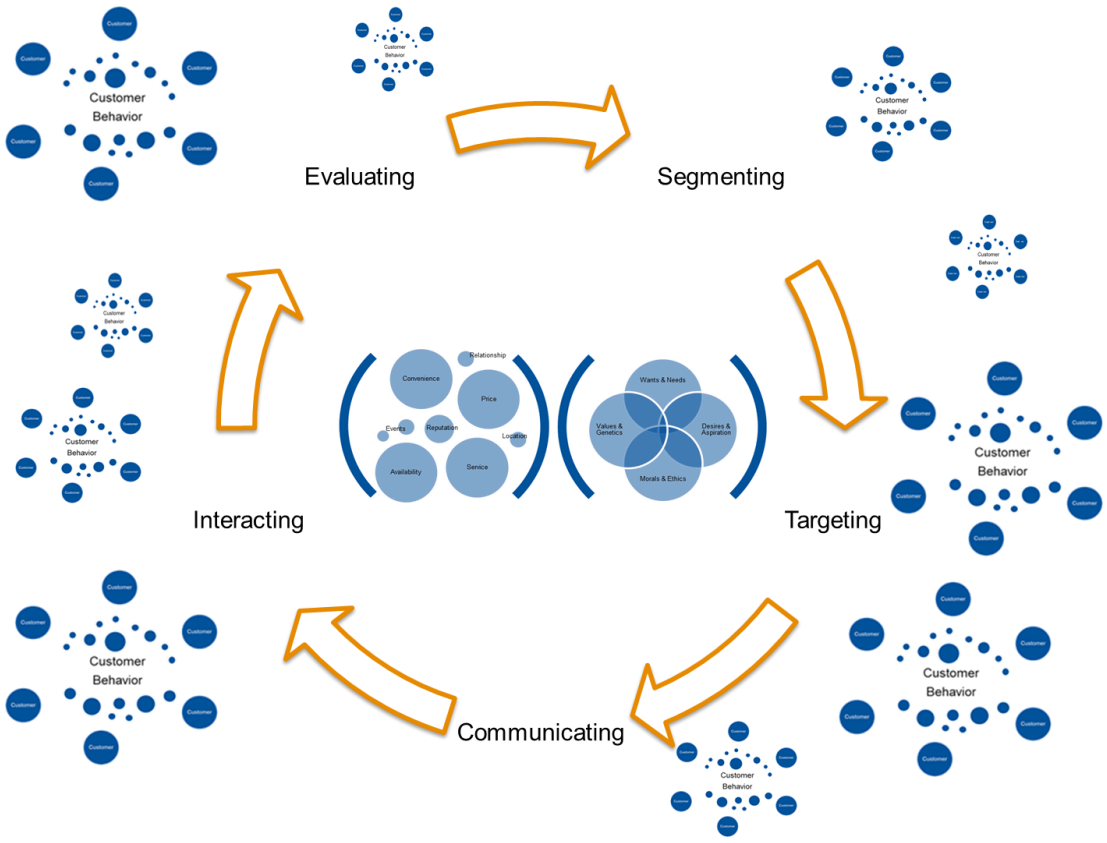
Capstone Project Report

Customer Behavior Prediction



Background

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/

Project Overview

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 analyze 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:

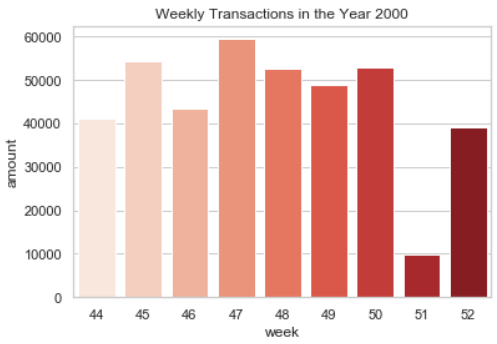
Inspired by the paper, Customer Shopping Pattern Prediction: A Recurrent Neural Network Approach by Hojjat Salehinejad and Shahryar Rahnamayan, the authors used a Recurrent Neural Network to predict customer loyalty values (R,F,M) using the **Ta Feng Grocery Dataset**. After doing some internet searching 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.

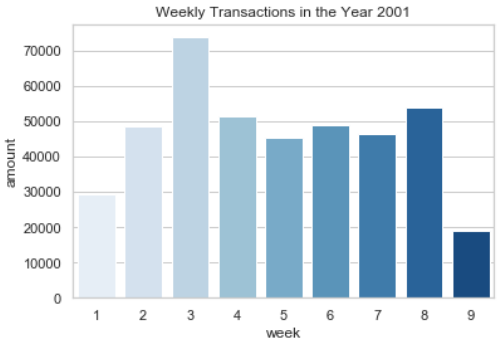
Data Wrangling

The data was separated into four separate datasets each representing a one month of data. For the most part the data was in a fairly clean format, the data was semi-colon separated and the first row of the dataset had garbage in it. Likely the first row was column names written in Chinese characters but obviously Jupyter Notebook and notepad couldn't make that distinction. I had to remove a number of white-spaces within the dataset in order to get everything to align correctly, but that was pretty easy to do, after the data was clean enough I was able to merge the data into one larger data-frame so that I could do some more work on the entire set of data. The first thing I did was I created some dictionaries with label mappings so that I could change the format of the 'age_group' column and 'pin_code' column easily without spending too much time on it. I also created the inverse of those dictionaries so that I may reverse those changes easily. Lastly, I created some operations to be able to encode dummy values on columns of my choosing, followed by changing as many of the columns into integer values as I could; since I knew that I would need to use the data for modeling or summarizing.

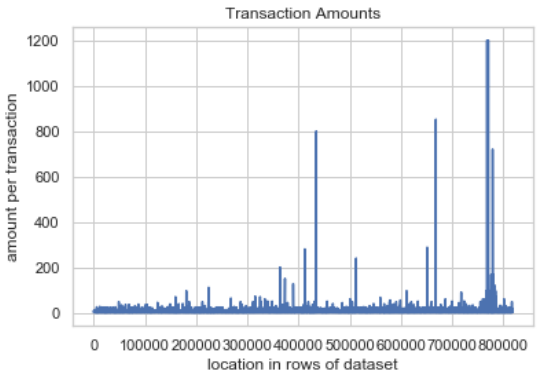
Exploratory Analysis:

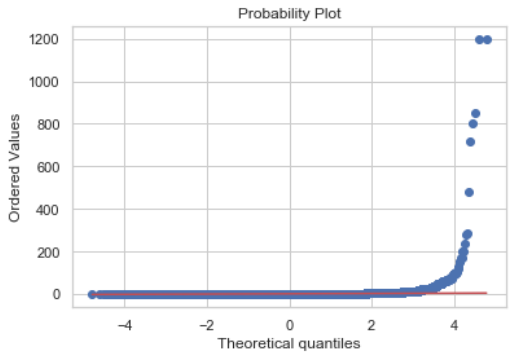
Once I had my data in a format that I was comfortable with I went about exploring the data itself. I started by graphing the columns and looking at their column statistics (mean, count, max, and percentiles) I created graphs where I could to help visualize the data.

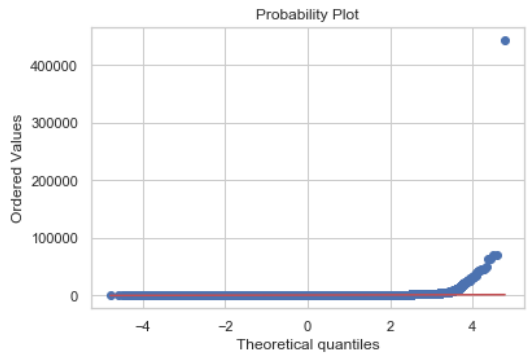




After looking at the differences in transaction counts by year I tested the hypothesis that perhaps the mean transaction counts of each year could be different from each other, and found the mean counts to not be statistically different from each other. I further looked at the distributions of 'Amount' and 'Sales Price' and found they had particularly odd behavior where there were dramatic spikes in activity for both purchasing 'Amounts' and 'Sales Prices'.





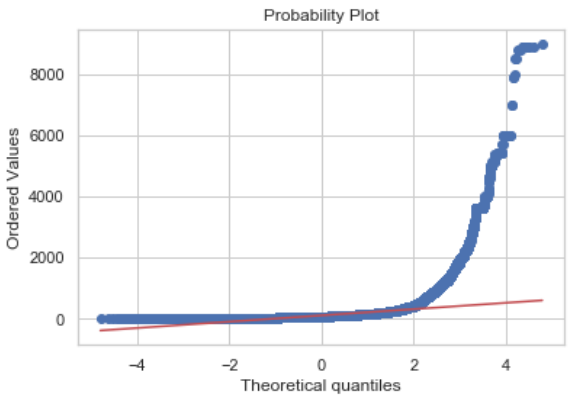


After further exploring these behaviors I found that there appeared to be a non-linear relationship with the two variables and decided that perhaps 'Sales Price' reflects the total transaction cost and not the individual 'Unit Price' that I initially believed.

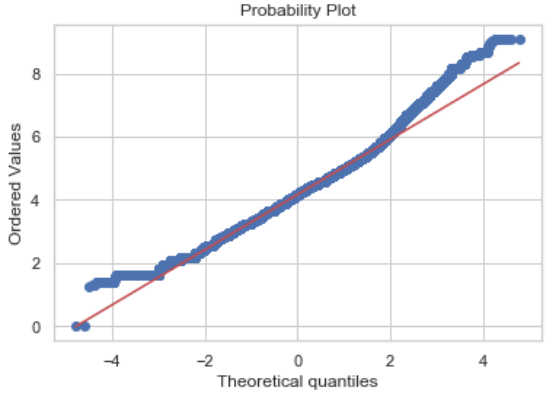
I tested both the 'Amount' variable and the 'Sales Price' variable and found neither variable were normally distributed; thereby, making it prohibitive to do a statistical test on their relationship. I finally finished off with dropping the extreme values from the data that I felt were unnecessarily large and

created a new variable in the data-frame ‘Unit Price’ that I felt most likely reflected the true cost of the items in the dataset.

Below: Unit Price



Log Transformed Unit Price



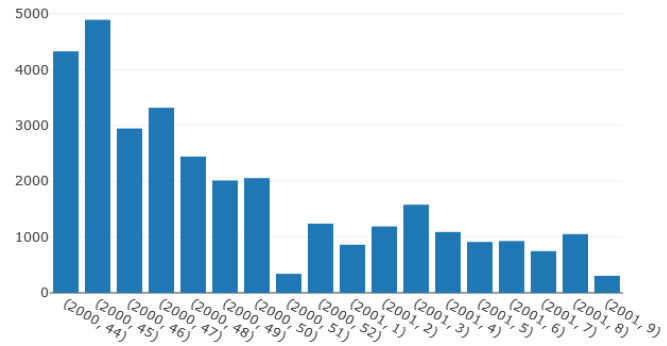
After investigating the effect of 'Amount' on 'Sales Price' I decided to remove a large portion of the most extreme of examples, which only comprise of less than 1% of the dataset. Doing so at least puts cost and purchase amount in a still exorbitant but understandable and identifiable range for my purposes; therefore that I limited the data to 40 items purchased in 1 transaction and about $1395.56 U.S. dollars (10000 yuan). Truth be told I can't think of the last time I purchased as many as 40 items and spent anywhere near 1000 dollars at a grocery store, but it’s not entirely unreasonable for the high-rollers. I think it is likely these data are either not actually from a 'grocery' store, or we might be dealing with data

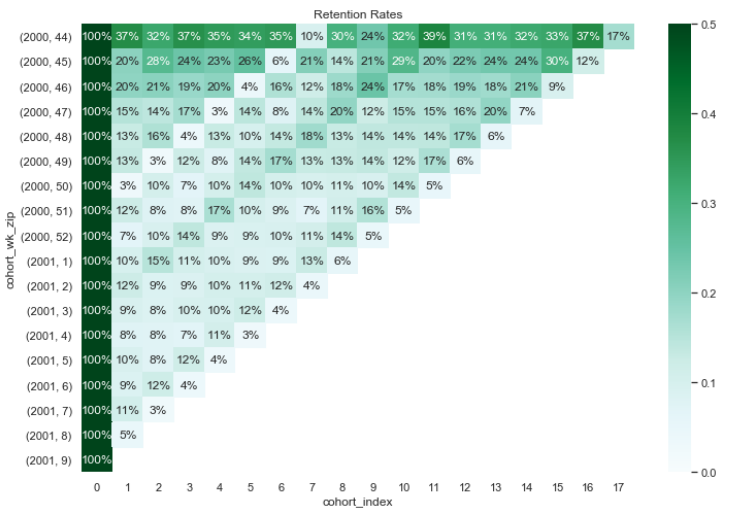
from an online system that automatically adjusts the prices according to the purchase amounts (supply/demand). Regardless of the reasons I would say that these data don't appear to me to follow the natural randomness of shoppers at a normal grocery store as one would expect to see. After doing statistical tests of both 'sales_price' and 'amount' we find that they indeed are *not* normally distributed; which also means that I cannot run a pearson-r correlation test to check for association.

Metrics

With my data exploration complete to my satisfaction my next goal was to help provide some valuable metrics about the customer purchasing behaviors in order to provide promotions and identify customers whom would be most valuable to the business or organization. In order to do so I grouped Customer Ids into cohorts of customers who made purchases around the same period of time and try to provide some valuable metrics for predicting purchasing behavior among those cohorts. Typically time cohorts would be separated by a yearly, monthly, or even daily basis for online customer groups. However, the nature of the data that I am working with doesn't lend itself to these former options very well because it is too short a time to separate the cohorts into yearly and monthly cohorts, and too long a time to separate into daily cohorts. So I chose to use a weekly length of time. Doing so added a certain amount of complication since a lot of time operations work well on the former time periods rather than the latter. I also chose to identify cohorts by the week that they enrolled from the start of the study to the end of the study (i.e. week 0 is November 1, 2000 and week 17 is February).

Below is a plot representing counts of customers for their first week purchasing an item. The X-axis is each year and week within the year, so (2000, 52) would be the last week of the year 2000.



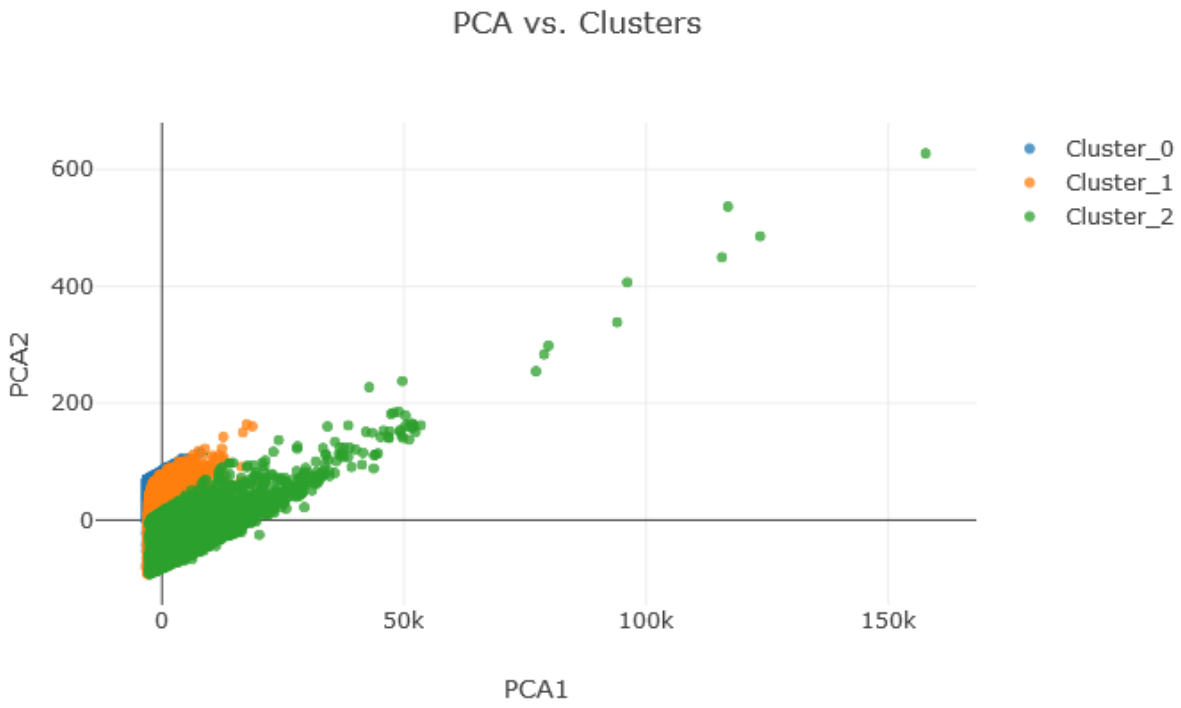


Once I had counts of cohorts I was able to create a retention table, or the counts of active customers whom purchased from one week to the next. I also created similar heat-maps for average transaction price as well as unit price. I found that typically customer cohorts bought anywhere from 1 to 1.5 items with the average cost being about 100 – 150 yuan. After creating the cohort groups the RFM metrics were my primary focal point for the project.

Behavioral customer segmentation is often based on these three foundation metrics:

1. Recency (R)
   - How many days since customer's last purchase (*the lower the better*) until the present
2. Frequency (F)
   - How many purchases the customer has done since their start of the time period
3. Monetary Value (M)
   - Measures how much the customer has spent since the start of the time period

With the RFM metrics calculated I was able to use the quantiles of each of these values and simply add them up in order to create their *RFM score*. The RFM score is a simplified way of segmenting customers based on their behavior in order to decide which customers are most valuable to the business. I further simplify this segmentation by classifying customers into only three metrics, gold, silver, and bronze level customers. In this way decision makers can draw conclusions about customers without having to get into the numbers. Finally, rather than using percentiles and quantiles to separate customers into recency, frequency, and monetary values I used K-means to separate them into semi-unsupervised groups. This has the added bonus of being easier to execute and letting the data decide the groups of customers. The downside of using this method is not specifically choosing the groups based on a simple metric that can be easily explained, and can be a bit more difficult to describe if needing to provide the solution to business partners.



Above we can see a 2-dimensional graph displaying the separation of clustered groups. After the groups were separated into groups I created a summary table of means among the groups.

Recency	Frequency	Monetary	Tenure	
mean	mean	mean	mean	count
60	7.1	946.2	2.4	12546
34.2	19.4	2512.7	55.2	11198
8.5	59.9	7546.2	91.8	8518

Interestingly, the graph above seems to show overlap among the groups while the table shows some clear differences in means. In order to test the assumption that the groups were meaningfully different from each other I chose to perform an F-test to see if the means were different from each other. And found them to be significantly different, or more to the point, not equal. I believe the chart and table seem to be a good illustration of why flattening data from multi-dimensional space to 2-dimensions to be misleading. Here we see a clear difference in clusters but that fact is not captured by the PCA graph.

Predicting RMF Values

Data Wrangling:

After importing the proper packages I downloaded the dataset from the exploratory data analysis 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.

Why am I calculating this yet again?

Because here I am doing something slightly different from my previous exploration of the RMF values, 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 previously.

So what this 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**.

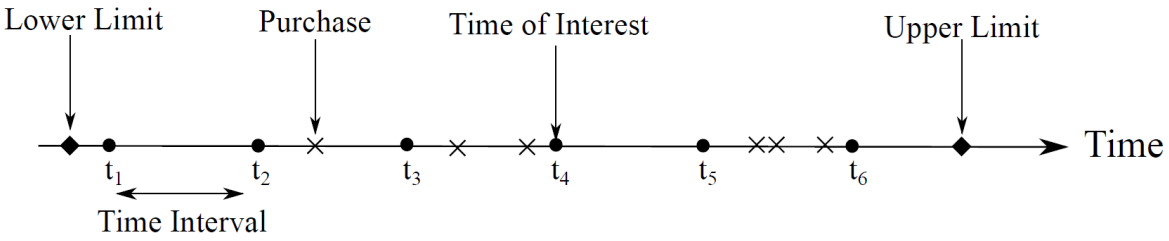


Fig. 3: A sample of shopper's behaviour during different time intervals.

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.

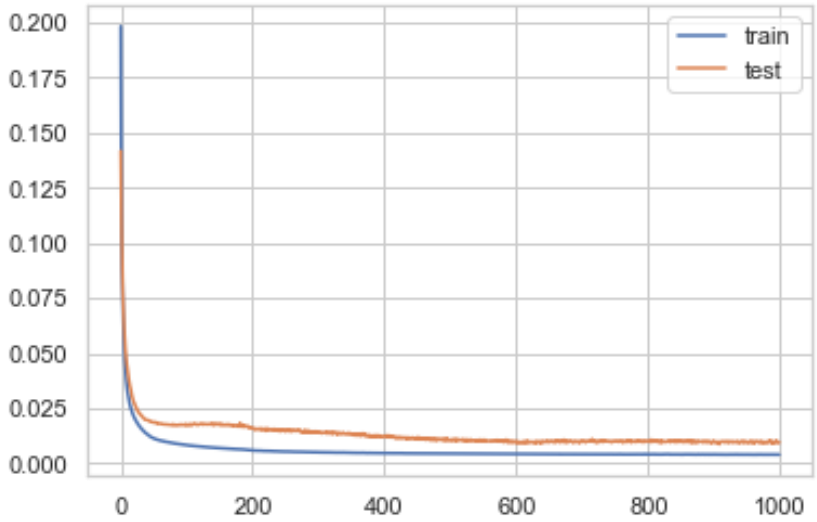
Hyperperameters

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

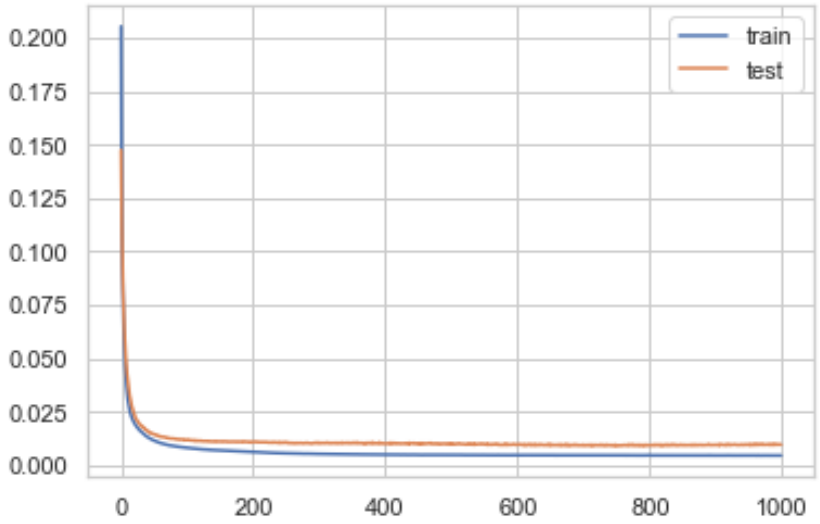
Full Model




```
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!