OnlineRetail Data Set Analysis

Martell Tardy

Pennsylvania State University

IE 575: Predictive Analytics

December 13, 2016

Overview: Summary of Report

In this report, the software used is RStudio 3.3.1. An introduction to the data set is presented in section 1 followed by a description of the RFM Analysis process to generate a target data set in section 2. In section 3 the clustering technique, k-means algorithm, is presented while section 4 explores forecasting techniques. Comparisons of the predictive methods are reviewed in section 5 followed by recommendations to the agency in section 6. A conclusion is offered in section 7 and lastly, references and an appendix stating my code from R in section 8.

Section 1: Introduction

An UK based non-store online retail agency is looking to improve future sales through the analysis of their retail data from 2010/12/01 to 2011/12/09. Through this analysis the online retail agency is hoping to gain insight into their top performing products, existing trends in their transaction history, and the forecasting analytics to improve sales moving forward. The data received from the agency contained transaction and demographic history for each purchase during the period of interest mentioned above. The transaction data set titled "OnlineRetail". This data set had 541909 observations and 8 variables consisting of InvoiceNo, StockCode, Description, Quantity, InvoiceDate, UnitPrice, CustomerID, and Country.

Section 2: Data Pre-Processing

In this section the RFM Analysis Process [1] was used to generate Recency, Frequency, and Monetary from the OnlineRetail data set. Moving forward, Recency or R should be understood as the number of days during the dates 2010/12/01 and 2011/12/09 for which an ID appears in the dataset. Frequency or F refers to the count of transactions an ID appeared during the dates 2010/12/01 to 2011/12/09. Monetary or M is the total value of transactions an ID made during the dates 2010/12/01 to 2011/12/09.

The result of this process, was a new data set titled “df” containing the ID(CustomerID), Date(InvoiceDate), Price(UnitPrice), Recency, Frequency, and Monetary from the original OnlineRetail data set. A sample of the *df* data set is shown in below in Table 1.

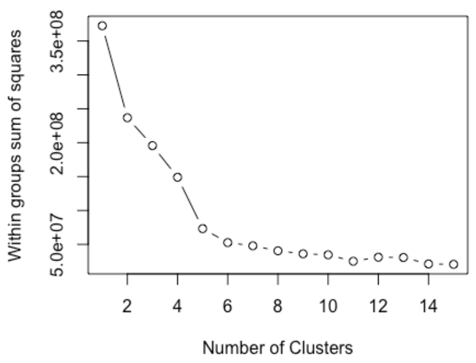
Table 1 – Sample of *df* Data Set

ID	**Date**	**Price**	**Recency**	**Frequency**	**Monetary**
12346	2011/01/18	1.04	324.791667	2	1.040000
12347	2011/12/07	1.45	1.791667	182	2.644011
12348	2011/09/25	1.25	74.833333	31	5.764839
12349	2011/11/21	7.5	17.791667	73	8.289041
12350	2011/02/02	2.1	309.791667	17	3.841176
12352	2011/11/03	1.65	35.833333	95	23.274737

Section 3: Clustering to Find Interesting Groups

Next, the data set “clear” was created containing R(Recency), F(Frequency), and M(Monetary Then a Scree Plot, which is a simple line segment plot that shows the fraction of total variance in the data. It is a plot, in descending order of magnitude, of the eigenvalues of a correlation matrix. In the context of factor analysis or principal components analysis, a scree plot helps the analyst visualize the relative importance of the factors, a sharp drop in the plot signals that subsequent factors are ignorable [2]. Using the sum of squared error (SSE) scree plot, the last drastic elbow in the plot concludes 5 clusters are indicated in this method (see Figure 2).

Figure 2 – Scree Plot



Then the K- Means algorithm was applied to the *clear* data set of 4372 observations. An ideal cluster will have a high Recency, Frequency, and Monetary. A cluster with a Frequency value >=3000, Recency value <=200, and a Monetary value>=2000 is considered a strong representation of the ideal transaction cluster. Function ggplot() was used to explore the visuals for each cluster.

Ggplot Visualizations

Figure 3 -Ggplot (Monetary ,Recency)

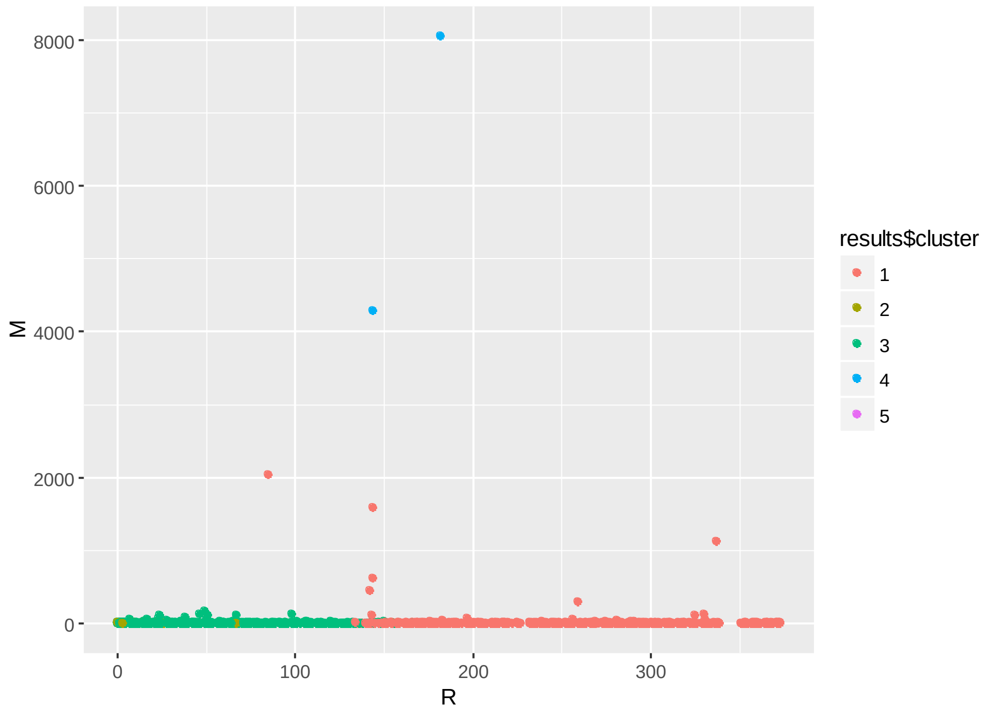


Figure 4 - Ggplot (Monetary , Frequency)

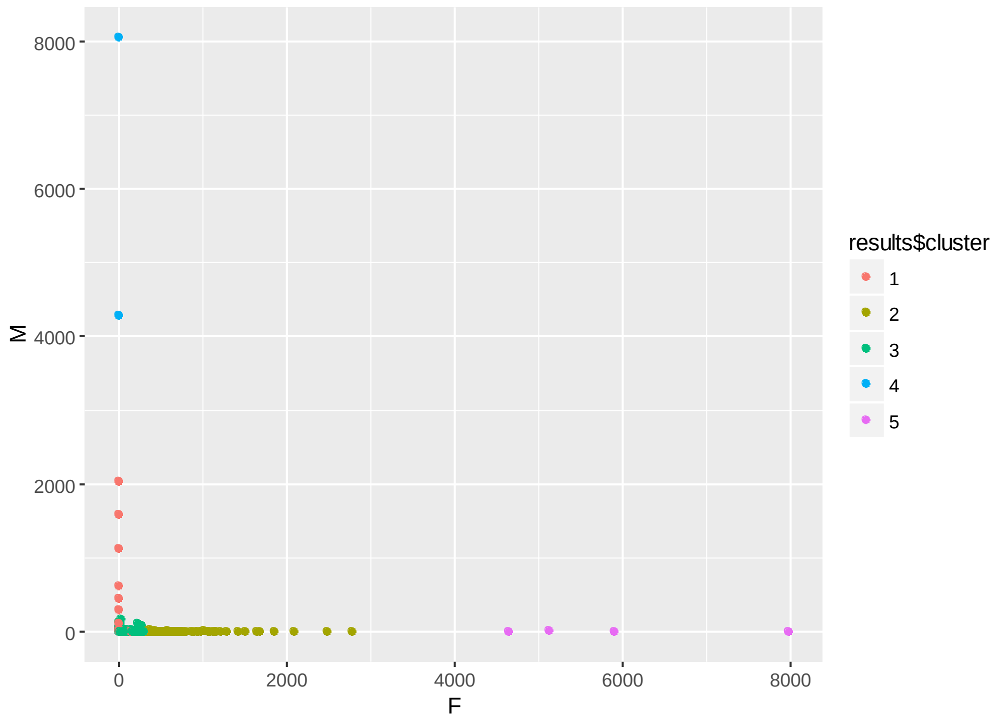
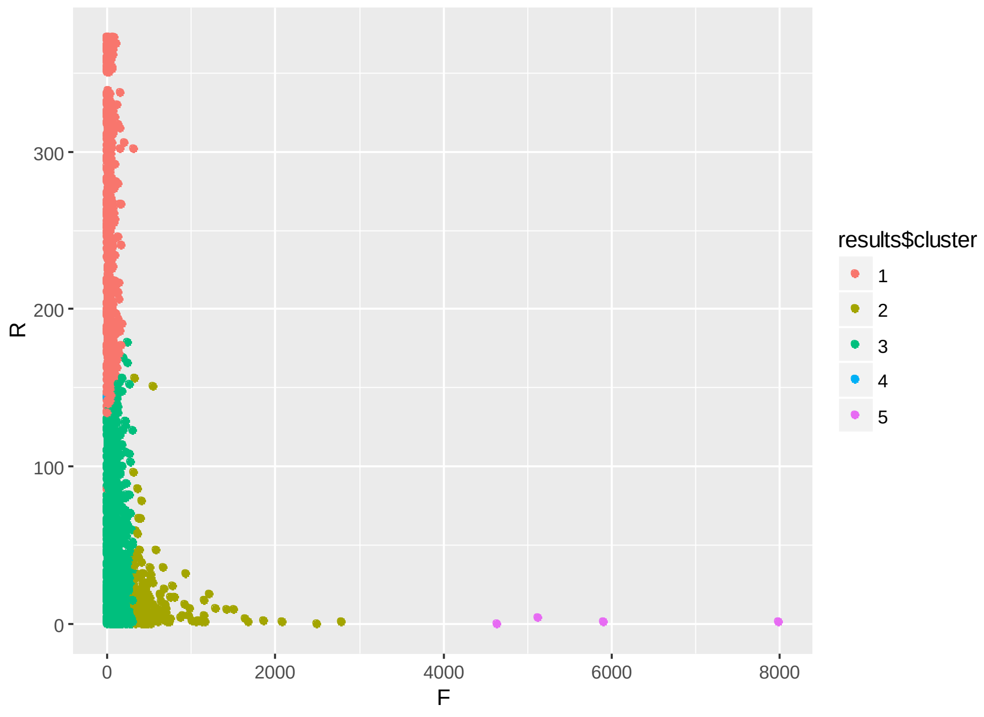


Figure 5 - Ggplot (Recency , Frequency)



Cluster Analysis

Cluster 1 relates to 1094 observations, composing 25% of all observations. Transactions in this cluster have occurred infrequently (F), predominately in the first half of the year (R), and have high monetary value (M).

Cluster 2 relates to 235 observations, composing 5.4% of all observations. Transactions in this cluster occurs infrequently (F), predominately in the second half of the time span (R), and have low monetary value (M).

Cluster 3 relates to 3037 observations, composing 69.5% of all observations. Transactions in this cluster have occurred infrequently (F), predominately in the second half of the time span, (R) and are moderate in monetary value (M).

Cluster 4 relates to 2 observations, composing .04% of all observations. Transactions in this cluster have occurred the least frequent (F), all in the second half of the time span (R), but have the highest monetary value (M).

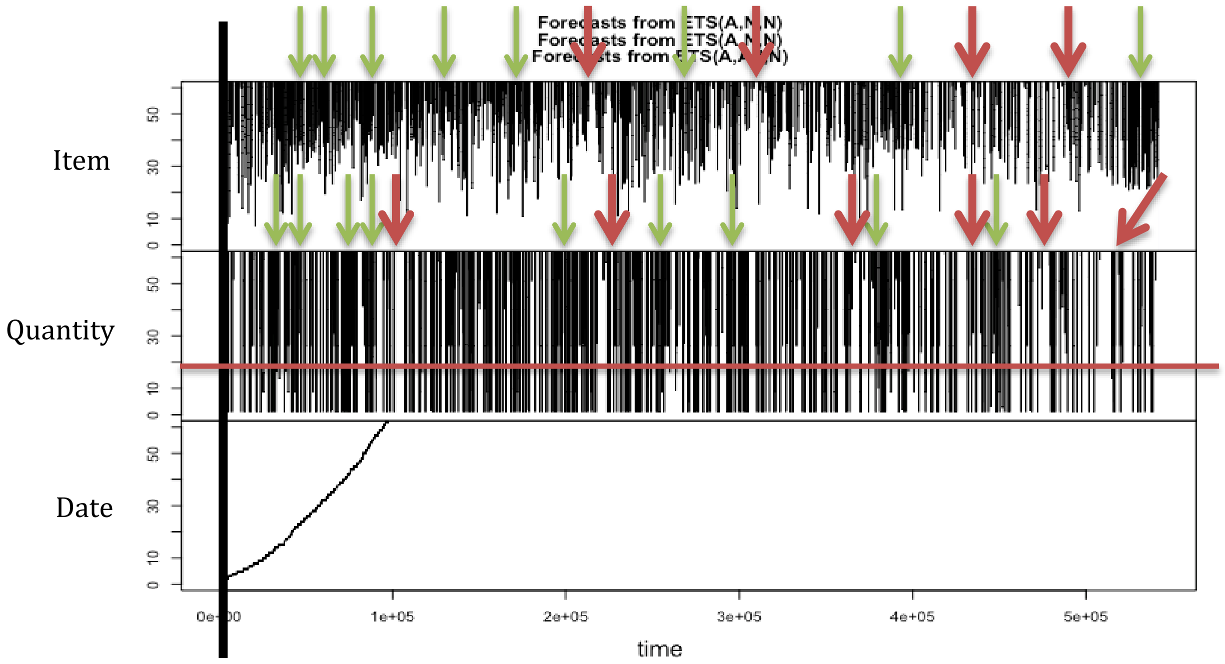
Cluster 5 relates to 4 to observations, composing .09% of all observations. Transactions in this cluster have occurred the most frequently (F), all within the second half of the time span (R), but do not have high monetary value (M).

Section 4: Predictive Sales Methods

Method 1: Moving Average Method

The moving average was calculated on data set "top_time" using function *forecast( ).* Data set *top_time* consisted of variables Item(StockCode), Quantity, and Date(InvoiceDate). The forecasting package in R was used to process the *top_time* data set and gain predictive insight into total revenue. First, a moving average, using the function ma( ), was executed to identify the trend direction and to determine support and resistance levels[3].

Figure 6 – Moving Average Method



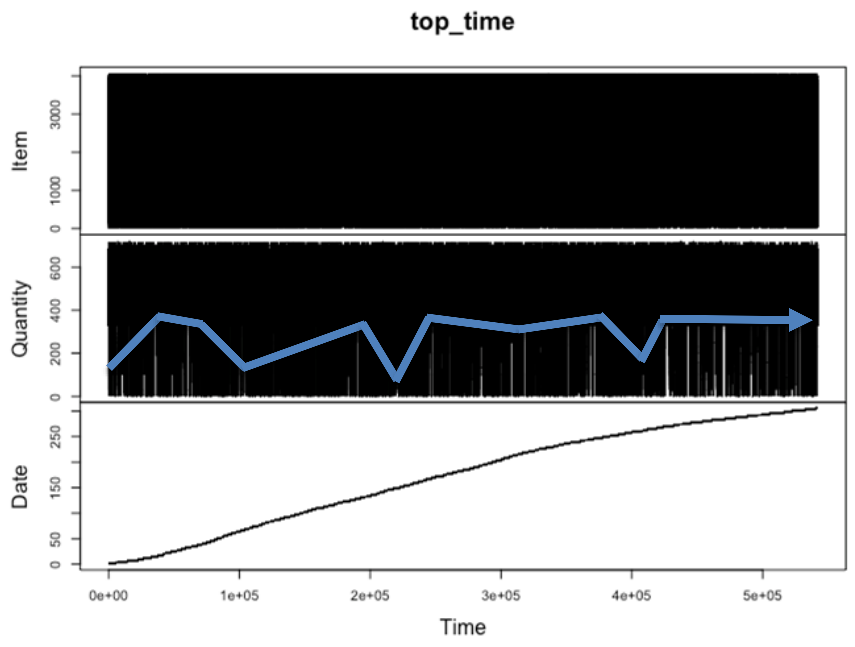
In Figure 6, a plot of the moving average is displayed. The time span is 12 months and 9 days; therefore, the time segments are reflective of approximately a 2-month duration along the x-axis. The variable Item in Figure 6 shows the frequency of each item (StockCodes) in transactions over the specified time frame. Instances where the variable Item appears high have been pointed to in green and when low in red. In these low instances it appears transactions of specific items occurred less frequently during the months of late April, July, September through October, and late November in 2011. From this, a trend of declining frequency of specific items (StockCodes) occurring in transactions during the Fall to Winter season should be explored further by the organization. Variable Quantity refers to the number of items per transaction. In the plot of variable Quantity in Figure 6 there is a trend line around 25, where consistent quantity of 25 items or more during each transaction occurs. This trend has been marked with a red line. For variable Quantity, there are noticeable instances of high low as well. These instances have been marked green (high) and red (low). In these low instances it appears the quantity of items per transaction occurred less frequently during the months of March, May, and continued in August through the remainder of 2011. It should be noted that the trend for low is especially noticeable toward the end of 2011 for variable Quantity. This could mean that the variable Quantity is affected by seasonal trends and therefore, should be explored further by the organization. Lastly, variable Date (InvoiceDate) shows a clear increase over time, which should be expected since the data for variable Date is of increasing days from the original start date of 2010/12/01.

Method 2: Time Span Analysis

To prepare for the Exponential Smoothing Method a Time Span Analysis was completed using function ts( ) on data set *top_time*.

Time Span Analysis

Figure 7 – Time Span [6]



In Figure 7, similar to Figure 6, a time span of 12 months and 9 days; therefore, the time segments are reflective of approximately a 2-month duration along the x-axis. The time span plot variable Item (StockCode) appears to show no drastic changes over this specific time span. However, variables Date and Quantity show clear variance over the time span. Variable Quantity has several noticeable spikes over time especially toward the beginning and end of the time span, both of which correspond to the winter months. Then a drastic drop off in February through June and then again in September. This trend is shown in blue in Figure 7. These variances may infer again that variable Quantity is affected by seasonal trends as seen in Figure 6. Variable Date shows a clear increase over time as well. This makes sense in reference to Figure 6.

Method 3: Simple Exponential Smoothing

Simple Exponential Smoothing is valuable when the data has no trend or seasonal patterns. Unlike a moving average, this technique gives greater weight to the most recent observations of the time series[3].

Figure 8 - SES

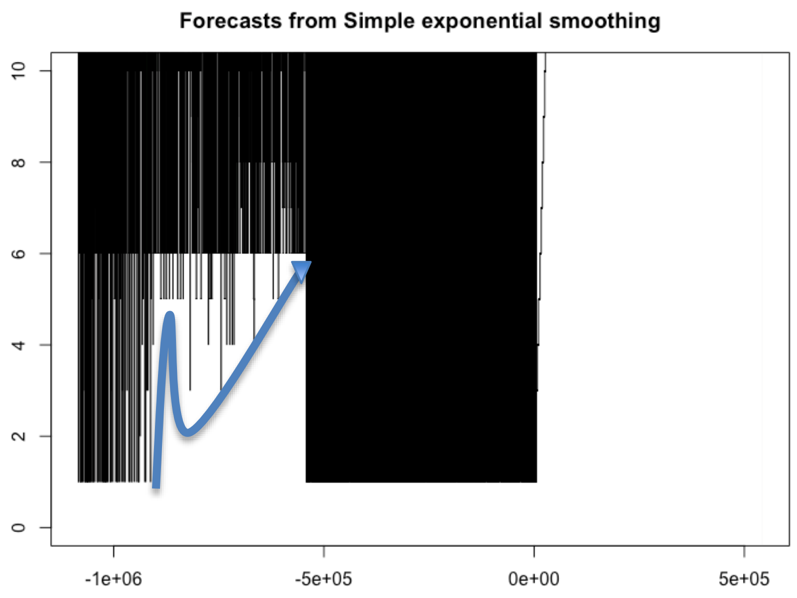
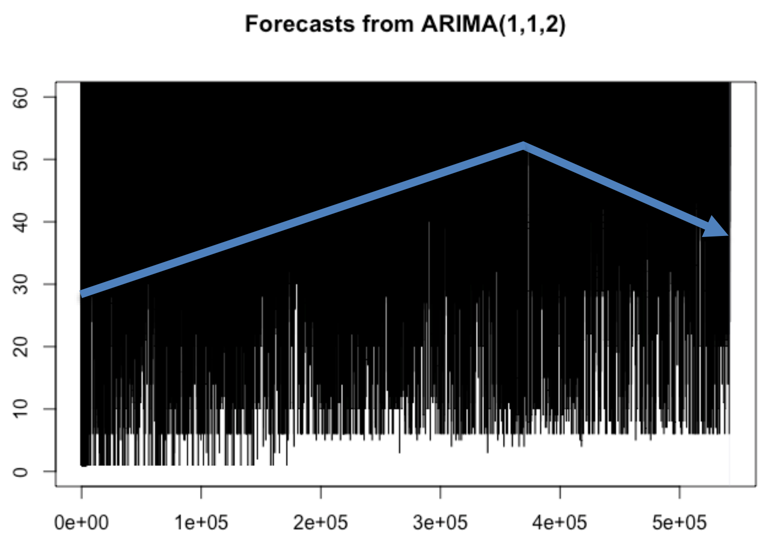


Figure 8 is a plot of the data from 2010/12/01 – 2011/12/09, showing a changing level over time, represented in blue but no obvious trending behavior based on variance.

Method 4: ARIMA

ARIMA stands for Auto-Regressive Integrated Moving Average. An ARIMA model can be viewed as a "filter" that tries to separate the signal from the noise, and the signal is then extrapolated into the future to obtain forecasts. The ARIMA forecasting equation for a stationary time series is a linear equation in which the predictors consist of lags of the dependent variable and/or lags of the forecast errors. The ARIMA model is displayed using "p" for the number of autoregressive terms, "d" for the number of non-seasonal differences needed for stationarity, and "q" for the number of lagged forecast errors in the prediction equation. [3]

Figure 9 - ARIMA



In Figure 9 an ARIMA (1,1,2) without constant equaling a damped-trend (leveling off) linear exponential smoothing is displayed. This model extrapolates the local trend at the end of the series but flattens it out at longer forecast horizons to introduce a note of conservatism, a practice that has empirical support [4]. The data is not stationary since a trend exists, represented by the blue line. Figure 9 shows a constant variance in its fluctuations over time. These fluctuations in the plot confirm the data is heavily seasonal and growing at a rapid rate. As a result, the ups and downs in the seasonality will potentially become more dramatic over time for this agency [5].

Section 5: Comparison of Predictive Methods

In section 4 the predictive methods Moving Average (MA), Time Span (TS), Simple Exponential Smoothing (SES), and ARIMA were applied to the *top_time* data set to examine the transaction history and examine forecast sales trends. After reviewing the results of the four models the MA method provided the most robust understanding and visualization of the data. In the MA model there are clear trends relating to each of the three variables and these observation are echoed in the other methods in a less complete fashion. The MA model, in comparison to the TS model, provides a noticeable trend of fluctuation for variables Item and Quantity, which is minimize in the TS model. The MA model, in comparison to the SES model, provides an understandable visualization of the data, while the SES model’s focus on variance does not. Lastly, the MA model, in comparison to the ARIMA model, both display the trend of increase, followed by a drastic dip toward the end of the time span. However, the MA model provides context of which variables are causing this trend and the recent increase in the MA model is only mildly apparent in the ARIMA model.

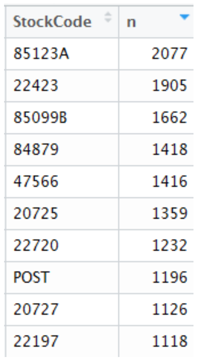
Section 6: Recommendations

The five clusters, created using the k-means algorithm, provide a snapshot of the transactions within the historical data provided for the time span 2011/12/09 to 2010/12/01. Through these five transaction clusters great examples of Recency, Frequency, and Monetary value are available. Cluster 1 provides a target data set of transactions with high Monetary value while Cluster 3 does so with Recency and Cluster 5 in Frequency. Cluster 2 and 4 display overlapping in their data with high Recency and Frequency for Cluster 2 and high Recency and Monetary value for Cluster 4. During this analysis an ideal transaction cluster displaying a Frequency value >=3000, Recency value <=200, and a Monetary value>=2000 was not found. However, using these five transaction clusters in further analysis, cluster demographic breakdown, and tracking of experimental strategies can be executed by the UK based non-store online retail agency to increase sales within each target group and potentially the development of the ideal transaction cluster.

In addition, the UK based non-store online retail agency should review more closely the findings from the Moving Average model. Observations pertaining to the time of year certain Items were seen in transaction and the Quantity of each transaction could be seen. This agency should review their pricing, product availability, and website optimization in the summer months since in each forecasting model there was a decline in the trend during this season.

Lastly, the top ten bestselling Items were calculated to provide context to the UK based non-store online retail agency on where the success in the products can be found. To find the top ten selling items library *dplyr* function *count( )* was applied to the OnlineRetail data set with focus on *StockCode*. From this application the following data set *top_ten* was created containing the top ten selling items which can be seen in Figure 10.

Figure 10 – Top 10 Sellers



StockCode	n
85123A	2077
22423	1905
85099B	1662
84879	1418
47566	1416
20725	1359
22720	1232
POST	1196
20727	1126
22197	1118

This agency should create a real-time rolling list of their best sellers and the relationship the five transaction clusters maintain with this list. Here is where this agency has a true opportunity to increase sales and see their experiential strategies to move the needle in real-time.

Section 7: Conclusion

The UK based non-store online retail agency was looking to improve future sales through the analysis of their retail data from 2010/12/01 to 2011/12/09. Through this report the online retail agency gained insight into their top performing products, trends in transaction history, and recommendation on ways to improve sales moving forward. The techniques used to provide this report were:

- RFM - *data pre-processing*
- SSE – *number of clusters*
- K-Means – *produce and analyze clusters*
- Ggplot – *visualize clusters*
- MA, TS, SES, ARIMA – *forecast and predict sales*
- Dplyr – *produce top ten selling items*

With the results and recommendations from this report, the UK based non-store online retail agency is well on their way to increasing further sales and customer retention.

Section 8:

References

[1]. "RFM Customer Analysis with R Language – dataapple.net." N.p., n.d. Web. 3 Dec. 2016.

[2]. "Scree Plots: Interpretation and Shortcoming – ba-finance-2013.blogspot.com." N.p., n.d. Web. 9 Dec. 2016.

[3]. "Basic Forecasting - r-bloggers.com." N.p., n.d. Web. 9 Dec.

[4]. "Introduction to ARIMA Models - People.duke.edu." N.p., n.d. Web. 9 Dec. 2016.

[5]. "Autoregressive Integrated Moving Average Models – forecastingsolutions.com." N.p., n.d. Web. 11 Dec. 2016.

[6]. "Time Series Analysis using R-forecast package – r-loggers.com." N.p., n.d. Web. 9 Dec.

Appendix: Code from R

Section2

```
> OnlineRetail <- read.csv("/Volumes/NO NAME/OnlineRetail.csv")//upload dataset to R

>   View(OnlineRetail)

> dat<- na.omit(OnlineRetail)//remove all NAs

> OnlineRetail=dat
> dim(OnlineRetail)

> OnlineRetail$InvoiceDate=format(as.POSIXct(strptime(OnlineRetail$InvoiceDate,"%m/%d/%Y
%H:%M",tz="")) ,format = "%Y/%m/%d" )// remove hours and minutes

> View(OnlineRetail)

> df <- as.data.frame(cbind(OnlineRetail[,7],OnlineRetail[,5],OnlineRetail[,6]))//dataframe Date,Price,ID

> names<-c("ID","Date","Price")

> names(df)<-names

> head(df)

> dim(df)

> dups<-df[!duplicated(df[,"ID"]),]//remove duplicate IDs

> dim(dups)

> startDate<-as.Date("2010/12/01","%Y/%m/%d")//date range to analyze

> endDate<-as.Date("2011/12/09","%Y/%m/%d")

> RFM<-function(df,startDate,endDate,ID="ID",Date="Date",Price="Price"){ //dataframe for RFM

+ df<-df[order(df[,Date],decreasing = TRUE),]

+ cleandf <- df[!duplicated(df[,ID]),]          //remove duplicates

+ Recency<-as.numeric(difftime(endDate,cleandf[,Date],units="days"))//get RECENCY

+ cleandf<-cbind(cleandf,Recency)// add days column

+cleandf<-cleandf[order(cleandf[,ID]),]//order(ID) to fit return

+ Freq<-as.data.frame(table(df[,ID]))//get FREQUENCY

+ Freq[,2]<-as.numeric(as.character(Freq[,2]))

+ Frequency<-Freq[,2]

+ cleandf <- cbind(cleandf,Frequency)

+df[,Price]<-as.numeric(as.character(df[,Price]))//get MONETARY

+ df[,ID]<-as.numeric(as.character(df[,ID]))

+ amount<-as.data.frame(tapply(df[,Price],df[,ID],sum))

+ Monetary<-amount[,1]/Frequency

+ cleandf<-cbind(cleandf,Monetary)

+ return(cleandf)

+ }

> df<-RFM(df,startDate,endDate)

> head(df)
```

Section3

```
> clear <- as.data.frame(cbind(df[,4],df[,5],df[,6]))
> names<-c("R","F","M")
> names(clear)<-names
> head(clear)
> wss <- (nrow(clear)-1)*sum(apply(clear,2,var))
> for (i in 2:15) wss[i] <- sum(kmeans(clear,
+ centers=i)$withinss)
> plot(1:15, wss, type="b", xlab="Number of Clusters",
+ ylab="Within groups sum of squares")
> results<-kmeans(clear,5) //creating 5 clusters
> results
> results$cluster<-as.factor(results$cluster)
> ggplot(clear, aes(R,M, color = results$cluster)) + geom_point( )//graphing clusters
> ggplot(clear, aes(F,M, color = results$cluster)) + geom_point( )
> ggplot(clear, aes(R,F, color = results$cluster)) + geom_point( )
```

Section 4

```
> top_time=as.data.frame(cbind(OnlineRetail[,2],OnlineRetail[,4],OnlineRetail[,5]))
> titles<-c("Item","Quantity","Date")
> names(top_time)<-titles
> install.packages("forecast")
> moving_average = forecast(ma(top_time,order = 2))//moving average method
> library("forecast", lib.loc="/Library/Frameworks/R.framework/Versions/3.3/Resources/library")
> moving_average_accuracy = accuracy(moving_average)
> moving_average;moving_average_accuracy
> plot(moving_average, ylim=c(0,60))
> lines(top_time)
> ts<-ts(top_time)//time span analysis
> head(ts)
> plot.ts(top_time)
> exp<-ses(top_time, 10, initial = "simple")//simple exponential smoothing method
> exp_accuracy = accuracy(exp,top_time$Date)
> exp; exp_accuracy
> plot(exp, ylim=c(0,10))
> lines(top_time)
> train=top_time$Item//ARIMA model
> arma_fit<-auto.arima(train)
> test=top_time$Date
> arma_forecast <- forecast(arma_fit, h = 100)
> arma_fit_accuracy <- accuracy(arma_forecast, test)
> arma_fit; arma_forecast; arma_fit_accuracy
> plot(arma_forecast, ylim=c(0,60))
> lines(top_time)
```

Section 6

```
> top<-dplyr::count(OnlineRetail,StockCode)//top ten best selling items
> View(top)
> top_ten<-dplyr::filter(top,n>1115)
> top_ten
> top_ten$StockCode=c(20725,20727,22197,22423,22720,47566,84879,85099,85123,9999)
> plot(top_ten)
```