|  |
| --- |
| CAPSTONE PROJECT SUBMISSION |
| Exploratory Analysis and Forecasting : Using Online Retail Data |
| RYERSON UNIVERSITY |

|  |
| --- |
| MD. FAHD AL GEORGY, STUDENT NO: 500809516  4-1-2019 |

Contents

[Introduction 3](#_Toc4968208)

[Literature Review 3](#_Toc4968209)

[Dataset 6](#_Toc4968210)

[Approach 7](#_Toc4968211)

[Step 1: Loaded packages in R 8](#_Toc4968212)

[Step 2: Loading and preprocessing of Data 8](#_Toc4968213)

[Step 3: Organizing Data 9](#_Toc4968214)

[Step 4: Exploratory Data Analysis 9](#_Toc4968215)

[Step 5: Time Series Analysis 15](#_Toc4968216)

[Discussion 19](#_Toc4968217)

[Conclusion 19](#_Toc4968218)

[Reference 19](#_Toc4968219)

[Appendix 20](#_Toc4968220)

[R Markdown 20](#_Toc4968221)

Table of figures

[Figure 1: Dataset organization 9](#_Toc4968222)

[Figure 2: Number of items and customers from different countries 9](#_Toc4968223)

[Figure 3: Country-wise selling trend 10](#_Toc4968224)

[Figure 4: Distribution of revisiting customers 10](#_Toc4968225)

[Figure 5: Selling trend per customer 11](#_Toc4968226)

[Figure 6: Number of sells/returns over time 11](#_Toc4968227)

[Figure 7: Relationship between selling and return over time 12](#_Toc4968228)

[Figure 8: Transaction trend over the time of the day 12](#_Toc4968229)

[Figure 9: Total earnings per day over the months 13](#_Toc4968230)

[Figure 10: Transaction history of the most sold items 13](#_Toc4968231)

[Figure 11: Item transactions over the time 14](#_Toc4968232)

[Figure 12: Overall sell trend 14](#_Toc4968233)

[Figure 13: Training and test data 15](#_Toc4968234)

[Figure 14: Model evaluation report 16](#_Toc4968235)

[Figure 15: Residual plot of training data 16](#_Toc4968236)

[Figure 16: Residual plot of testing data 17](#_Toc4968237)

[Figure 17: Actual vs predicted earning for test data 17](#_Toc4968238)

[Figure 18: Model's prediction of earning from sell 18](#_Toc4968239)

[Figure 19: Forecast with confidence interval 18](#_Toc4968240)

[Figure 20: Forecast with Prophet model 19](#_Toc4968241)

Exploratory Analysis and Forecasting

# Introduction

The Online retail market has been the fastest growing shopping trend worldwide during the last decade. All famous large retailers in the world have opened up their online shopping portals to cater to the need of their customers. In the age of globalization, online retailing has expanded the customer base from being limited to a boundary of any specific geographic location. For example, customers in the UK now can search, verify, judge and order any quality product from the USA. However, for both large and small retailers, this has created an immense pressure and put them in fierce completion to maintain market share let alone expand. Hence, Data Analysis to analyze customers' purchasing behavior, understanding the shopping pattern, having an idea on their likes, dislikes have become increasingly important to retain success.

On the basis of the above, I am proposing a project that aims at performing an exploratory analysis on an *online retail dataset*. The objective of this analysis would be to predict the trends of online purchasing, forecast expected revenue trend and identify underlying relations among the products that customers tend to buy together.

# Literature Review

Retail analytics focuses on providing insights related to sales, inventory, customers, and other important aspects crucial for merchants’ decision-making process. The discipline encompasses several granular fields to create a broad picture of a retail business’ health, and sales alongside overall areas for improvement and reinforcement. Essentially, [retail analytics](https://www.sisense.com/solutions/retail/) is used to help make better choices, run businesses more efficiently, and deliver improved [customer service analytics](https://www.sisense.com/solutions/customer-service/). Moreover, companies use this analytics to create better snapshots of their target demographics. By harnessing [sales data analysis](https://www.sisense.com/solutions/sales/), retailers can identify their ideal customers according to diverse categories such as age, preferences, buying patterns, location, and more.[[1]](#footnote-1)

There are several excellent retail analytics examples that are relevant to a variety of companies. One of the biggest benefits the field delivers to companies is optimizing their inventory and procurement. Thanks to [predictive tools](https://www.sisense.com/glossary/predictive-data-analytics/), businesses can use historical data and trend analysis to determine which products they should order, and in what quantities instead of relying exclusively on past orders. In addition, they can optimize inventory management to emphasize products customers need, reducing wasted space and associated overhead costs. Apart from inventory activities, many retailers use analytics to identify customer trends and changing preferences by combining data from different areas. By merging sales data with a variety of factors, businesses can identify emerging trends and anticipate them better. This is closely tied to marketing functions, which also benefit from analytics. Companies can harness retail analytics to improve their marketing campaigns by building an improved understanding of individual preferences and gleaning more granular insights. By blending demographic data with information such as shopping habits, preferences, and purchase history, companies can create strategies that focus on individuals and exhibit higher success rates.[[2]](#footnote-2)

Performing sales trend analysis gives a firm valuable insight into the inner-workings of its business. Merchants use their data to make informed decisions like when to raise or lower prices on your products. These decisions shouldn’t always be a “gut” feeling. While gut can sometimes be reliable, it should not be the only decision-making tool. When looking for trends or patterns in sales data, we can determine both opportunities and potential problems. We can track if a particular product is increasing or decreasing in sales. If it’s declining, we can make timely decisions such as to cut prices, market more, or discontinue the product. If an item is selling off the shelves, we can be sure to stock inventory accurately across channels. Sales trend analysis also helps us determine if we are meeting our sales goals by providing an easy, measurable way to track our progress. We actually get to know if we increased sales from last year and by what percentage. If a goal was not met, we can drill down to sales of a specific product or location to see what is stopping us. All retailers should have the ability to become data-driven businesses. With the right capabilities, we can have confidence in the decisions we make because they are backed by our own data.[[3]](#footnote-3)

Relying on retail analytics and hard data rather than guesswork enables us to make smarter decisions toward higher profits, better customer satisfaction, and having a more awesome store overall**.** The good news is that it looks as though many players in the retail industry have already recognized the importance of data. [[4]](#footnote-4)

[A survey by Alteryx and RetailWire](http://www.alteryx.com/shopper-insights-insider-perspectives-infographic)of nearly 350 retailers and brand manufacturers found that 81% of respondents say they gather shopper insights and 76% consider insights to be critical to their performance. The bad news is that while many merchants are collecting data, most aren’t using it effectively. According to the study, only 16% consider themselves experts when it comes to data harnessing, while 24% and 60%, respectively, describe themselves as “newbies” and “getting there.”[[5]](#footnote-5)

Review of similar work done in retail analytics:

Sven Anders and Anke Moeser conducted research on organic and conventional fresh beef products in the Canadian retail market. In their paper published on 13 August 20108, Sven Anders et all. Mentioned that they applied sets of weekly retail and household scanner data to estimate consumer demand for selected organic and conventional fresh beef products in the Canadian retail market. The main contribution of the study stems from the application of a two‐stage procedure that provides new and deeper insight into consumers' responses to changing retail environment and pricing for organic and conventional meat products. The combined knowledge of point‐of‐sale consumer behavior for value‐based products, such as organic products, and distinct socio‐demographic profiles of buyers vs. non‐buyers of meat is especially interesting for retail managers and meat industry stakeholders. First, household meat consumption patterns are investigated based on household scanner data that track household's meat purchases in the period 2006–2007. The second step of analysis then involved the estimation of an almost ideal demand system for selected organic and conventional fresh beef products using retail scanner data for the period 2000–2007. The introduction of greater selections in organic product lines across mainstream supermarkets in Canada in response to consumer health concerns is expected to spur retail competition in an otherwise saturated Canadian retail market. The analysis of socio‐demographic profiles in beef consumption using individual household's purchase data revealed that besides regional differences in preferences, household size and resource characteristics are major determinants of point‐of‐sale beef purchase decisions. The demand system results indicated that organic beef is highly dependent on price and expenditures, whereas demand for conventional beef is mostly driven by income, habits and ‘typical’ Canadian seasonal beef consumption patterns.[[6]](#footnote-6)

Mark Harman and et all in their paper introduced app store mining and analysis as a form of software repository mining. Unlike other software repositories traditionally used in MSR work, app stores usually do not provide source code. However, they do provide a wealth of other information in the form of pricing and customer reviews. Therefore, Harman et all supposedly used data mining to extract feature information, which then combined with more readily available information to analyze apps' technical, customer and business aspects. The approach was applied to the 32,108 non-zero priced apps available in the Blackberry app store in September 2011. Results showed that there is a strong correlation between customer rating and the rank of app downloads, though perhaps surprisingly, there is no correlation between price and downloads, nor between price and rating. [[7]](#footnote-7)

The National Retail Data Monitor (NRDM) is a public health surveillance tool that collects and analyzes daily sales data for over-the-counter (OTC) health-care products. NRDM collects sales data for selected OTC health-care products in near real-time from >15,000 retail stores and makes them available to public health officials. NRDM is one of the first examples of a national data utility for public health surveillance that collects, redistributes, and analyses daily sales-volume data of selected health-care products, thereby reducing the effort for both data providers and health departments.[[8]](#footnote-8)

# Dataset

The ‘Online Retail’ data set contains the transactions that took place between 01/12/2010 and 09/12/2011 for a registered non-store online retail in the UK. The main product line of the company is unique all-occasion gift items. Many customers of the company are wholesalers.

**Attribute Information:**

We have in total of 8 attributes with 5 nominal attributes.

InvoiceNo: Invoice number. Nominal, a 6-digit integral number uniquely assigned to each transaction. If this code starts with the letter 'c', it indicates a cancellation.

StockCode: Product (item) code. Nominal, a 5-digit integral number uniquely assigned to each distinct product.

Description: Product (item) name. Nominal.

Quantity: The quantities of each product (item) per transaction. Numeric.

InvoiceDate: Invoice Date and time. Numeric, the day and time when each transaction was generated.   
UnitPrice: Unit price. Numeric, Product price per unit in sterling.

CustomerID: Customer number. Nominal, a 5-digit integral number uniquely assigned to each customer.

Country: Country name. Nominal, the name of the country where each customer resides.[[9]](#footnote-9)

# Approach

## Step 1: Loaded packages in R

In the beginning, the required packages in R were loaded. The environment for subsequent analytical procedures became ready. The packages were:

|  |  |
| --- | --- |
| broom | To prepare a tidy frame of statistical objects those used later by tidyverse and tidyquant |
| caret | To perform regression and classification on our dataset |
| ggcorrplot | To visualize correlation among the features of our dataset |
| grid | Convert visualization among different grid systems |
| gridextra | Visualize multiple grid base plots |
| matrixStats | Perform matrix operations on our dataset |
| modelr | Integrate modeling into the pipeline of manipulation and visualization of our dataset |
| prophet | Forecast time series |
| tidyverse | Provide a useful set of libraries that aid in analysis and visualization |
| timetk | Forecast time series |
| tidyquant | Provide better visualization of different plots |

## Step 2: Loading and preprocessing of Data

We have loaded the dataset "Online Retail.csv" and extract some information from the "InvoiceDate", "Quantity" and "UnitPrice" features for analysis purpose. The "InvoiceDate" column contained the date in a year-month-date format composed with the transaction time in hh:mm: ss format. From this, we added 4 columns named sell\_date (year-month-date), day\_of\_week (e.g. Mon, Tue, etc.), sell\_time (hh:mm:ss) and month (1 to12). We further added one column named ‘earning' with the corresponding row values of Quantity \* UnitPrice. We have canceled transactions and returned item data in our dataset. The canceled transactions are marked by ‘C' at the beginning of the invoice number and placed a negative number in the ‘Quantity' column. Based on these values of ‘Quantity' feature, we have added one extra column named ‘sell\_status' which indicates the item was sold or not.

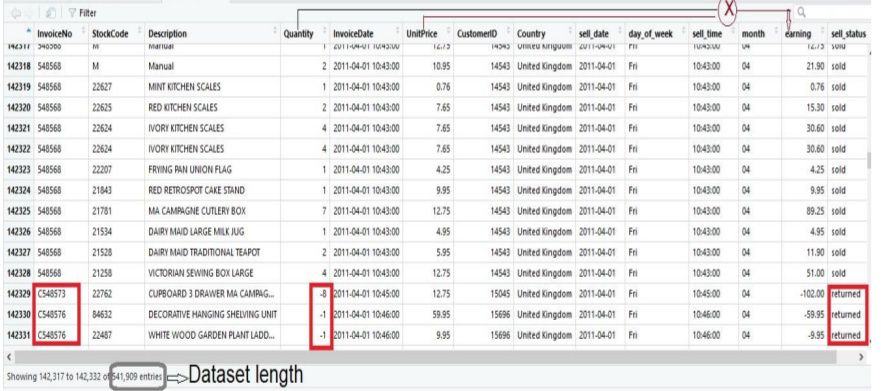


Figure 1: Dataset organization

## Step 3: Organizing Data

We have also checked for missing values and have spotted missing values in product description and customer ID column. As the missing values of these two columns will have a negligible impact on forecasting, we have tried to overlook those.

## Step 4: Exploratory Data Analysis



Figure 2: Number of items and customers from different countries

As from figure 2, we have a total of 4070 unique items that were purchased by 4373 customers from 38 different countries. We might want to see that the customers from which countries performed most transactions and thus we plotted figure 3.

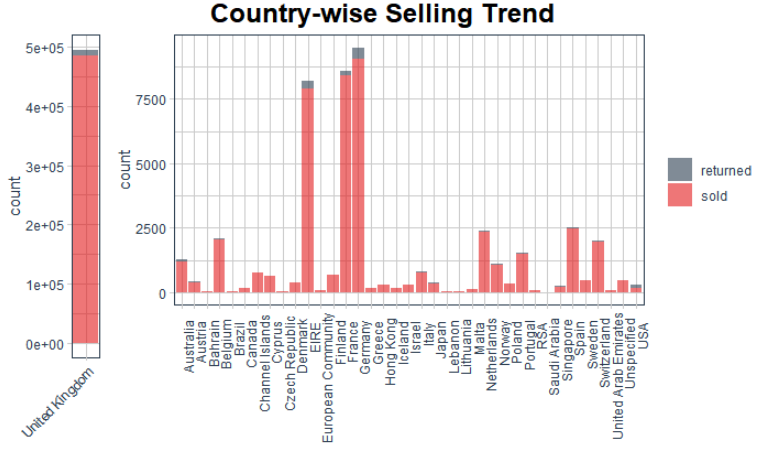


Figure 3: Country-wise selling trend

Figure 3 depicts that the largest product sold in the UK, most probably because the retail company was UK based. We have also a high number of sold products in Germany, France, and Ireland. As we have around 542K transactions performed by 4373 customers, surely there will be many revisiting customers, and figure 4 shows us the ratio of one-time customers and revisiting customers.

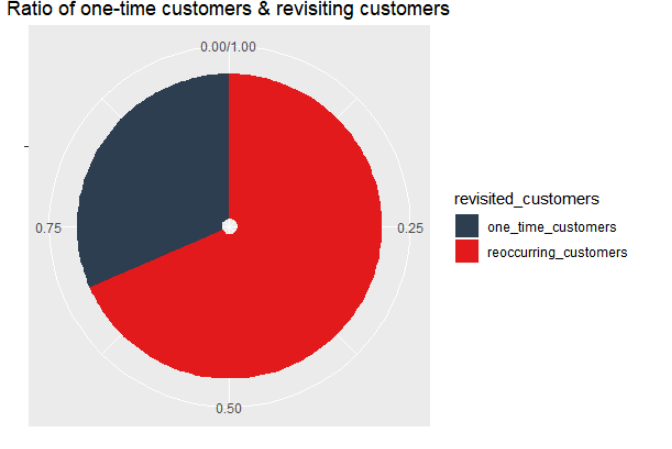


Figure 4: Distribution of revisiting customers

Figure 5 depicted that the mean earning per customer was just above 500 GBP. The number of different items bought by each customer was approximately 30 and the mean quantity of those items were nearly 300.

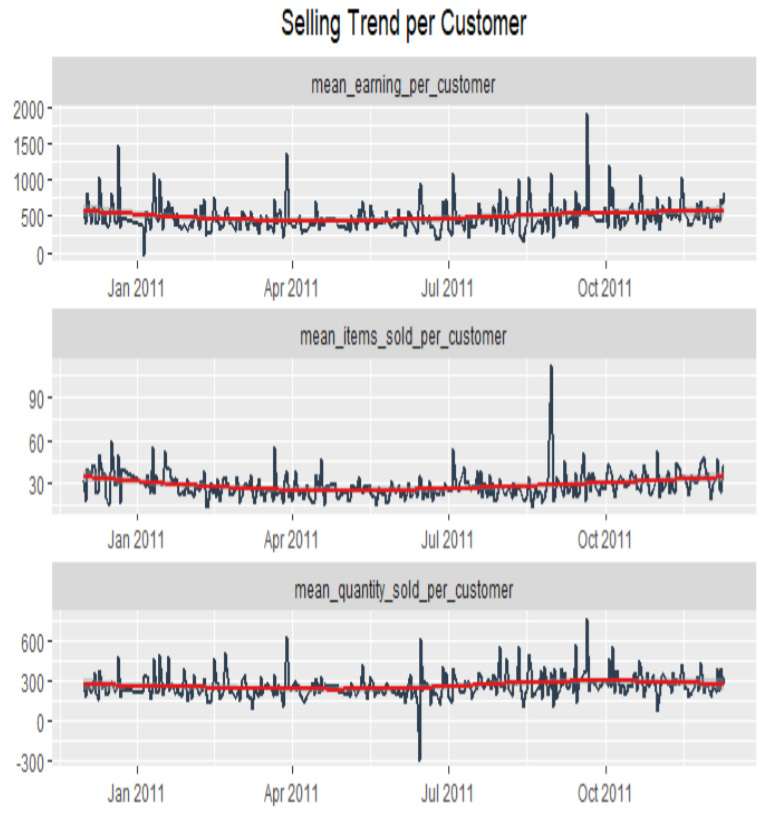


Figure 5: Selling trend per customer

As we have many canceled transactions in our dataset, we might want to see the number of sells and return over time and the relationship among them.

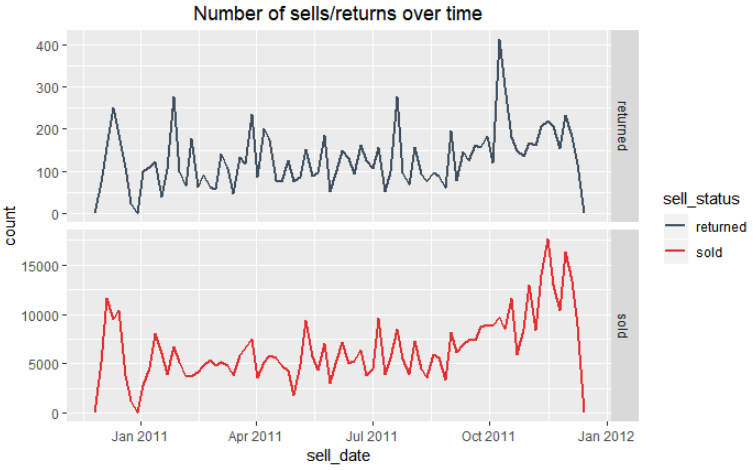


Figure 6: Number of sells/returns over time

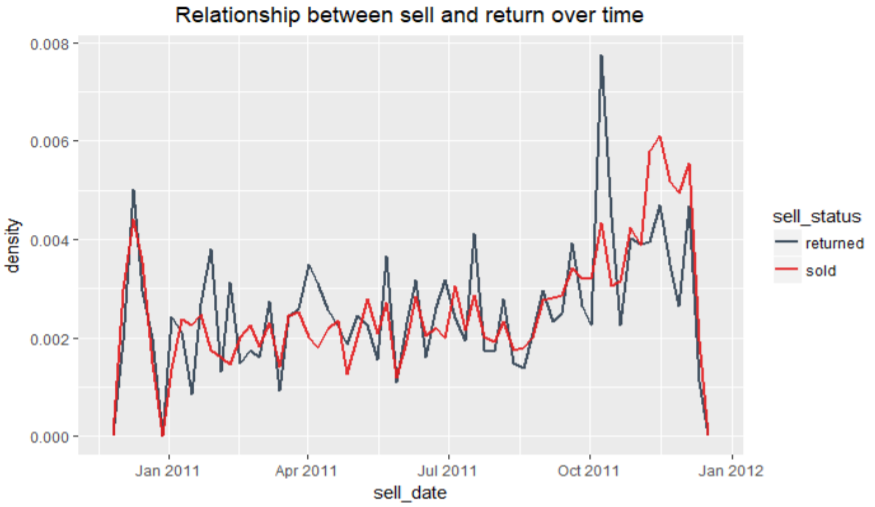


Figure 7: Relationship between selling and return over time

Figure 8 depicted that most transactions occurred between 10 am to 5 pm. There were no transactions between 10 pm to 6 am. The highest number of transactions occurred between the ends of September 2011 to the 9th of December 2011.

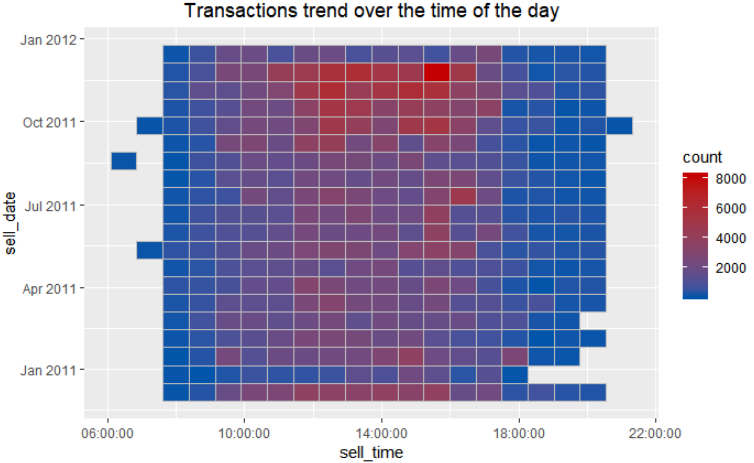


Figure 8: Transaction trend over the time of the day

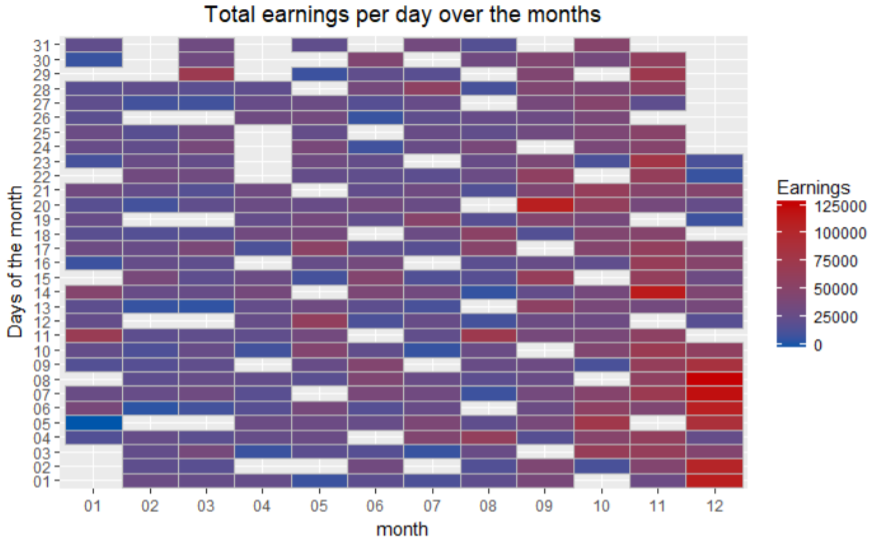


Figure 9: Total earnings per day over the months

From figure 9 depicted the highest earning days over time. The sell increased during November and reached to the highest pick in the first week of December 2011. Next, we would like to see the transaction history of the most sold items of our dataset, depicted in figure 10.

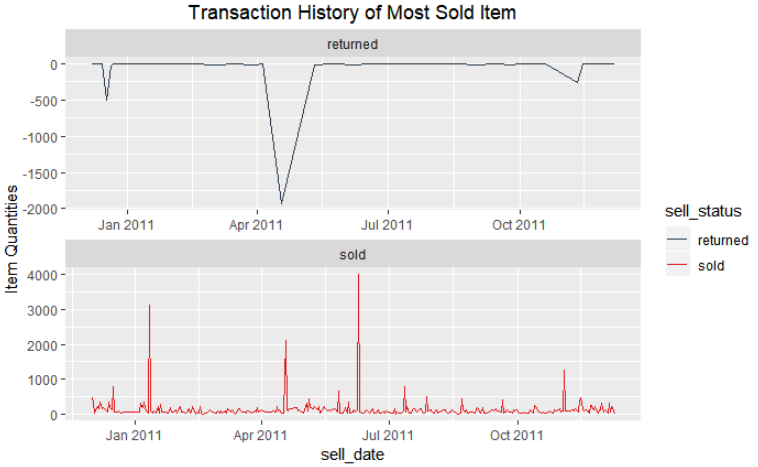


Figure 10: Transaction history of the most sold items

By figure 11, we spotted that the transactions of items increased after the end of July and in December the number of items was sold on average were reached approximately 1500.

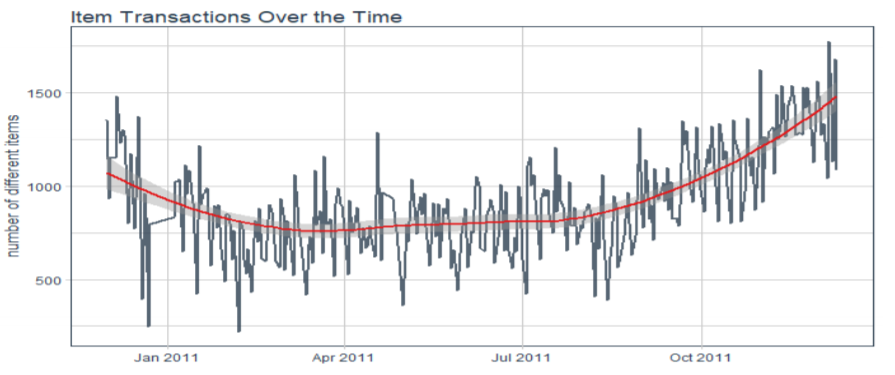


Figure 11: Item transactions over time

The overall earning by sell over the time did not fall below 4000 GBP and the earning ranged between 4K to 7.5 on an average, as depicted in figure 12. There is a hike and down in earning throughout the year. We have noticed that at the end of December 2010 there were less earning for a couple of days. The sell earning remained between 500 to 12000 throughout the year with one exception at the end of December 2011. Surprisingly, the sell earning reached up to 20K in that time.

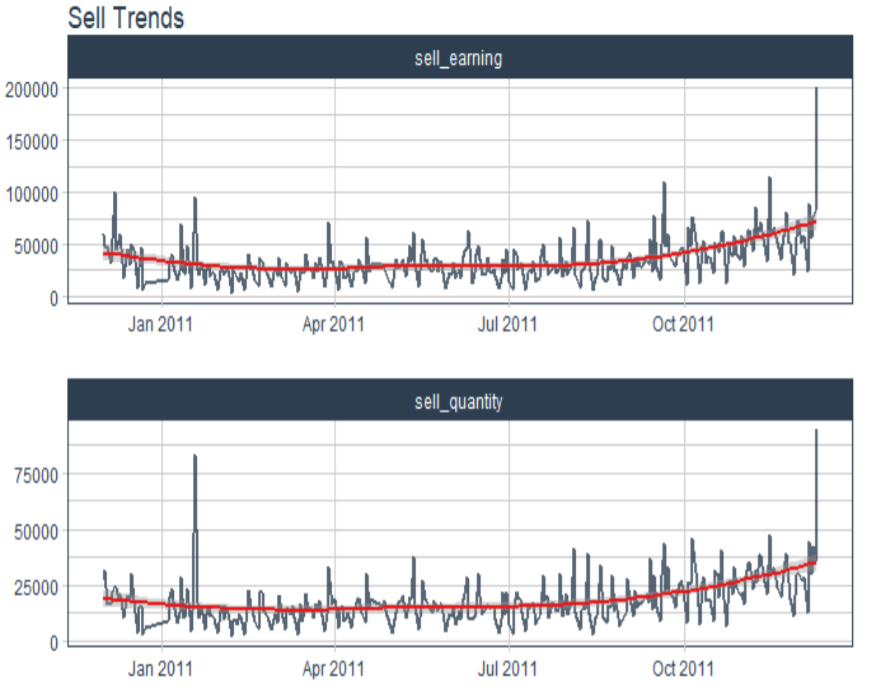


Figure 12: Overall sell trend

The overall quantity of items sold was ranged in between 150 to 50000. We have noticed 2 exceptions. One occurred at the end of January 2011 when the item quantity of sold items reached above 75000 and the second one was at the end of the graph showed in figure 12, which was the month December in 2011 where the quantity reached nearly 100K.

## Step 5: Time Series Analysis

We have decided to forecast on our sell earnings by using the available data. As a part of this process, we have divided our dataset into training and testing set. We have used data from 1 st of December 2010 to 30th September 2011 to train our model. Data from 1st October 2011 to 9th December 2011 has kept for the testing purpose, as figure 13 shows.

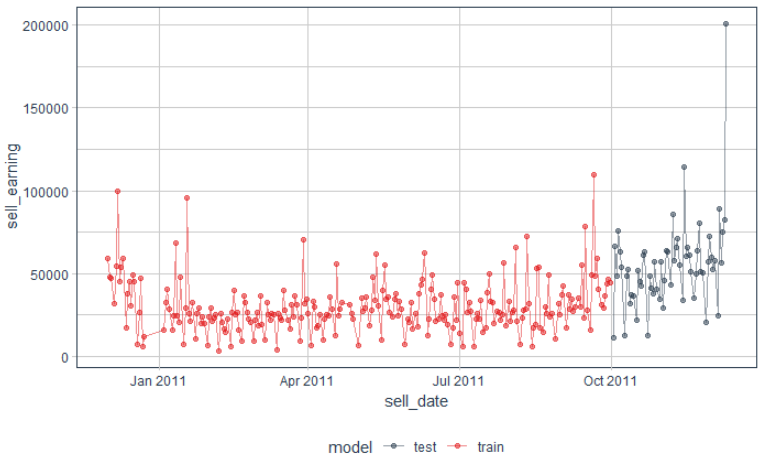


Figure 13: Training and test data

Using the training data, we have trained our model and evaluated the performance. Our p-value and standard error reflects the performance of our model in figure 14.

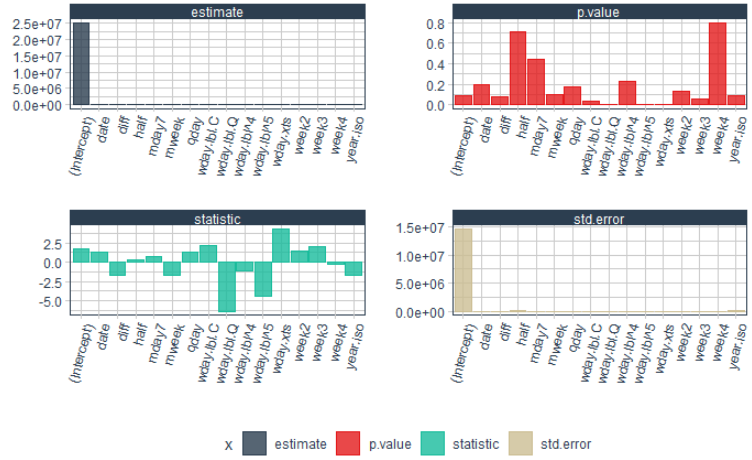


Figure 14: Model evaluation report

We have used a simple linear model for this time series analysis where the response variable is ‘sell\_earning'. Figure 15 and figure 16 shows the working ability of our model. Both figures (15 &16) shows that the difference between expected and predicted has not deviated a lot.

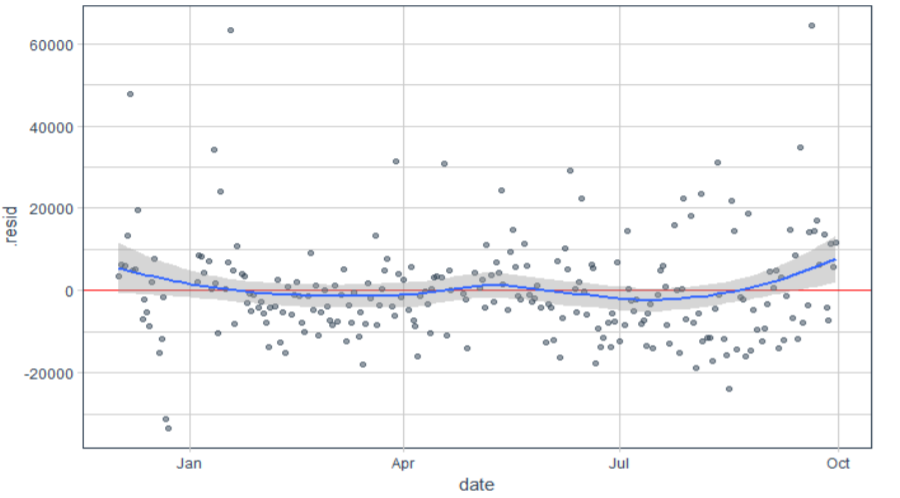


Figure 15: Residual plot of training data

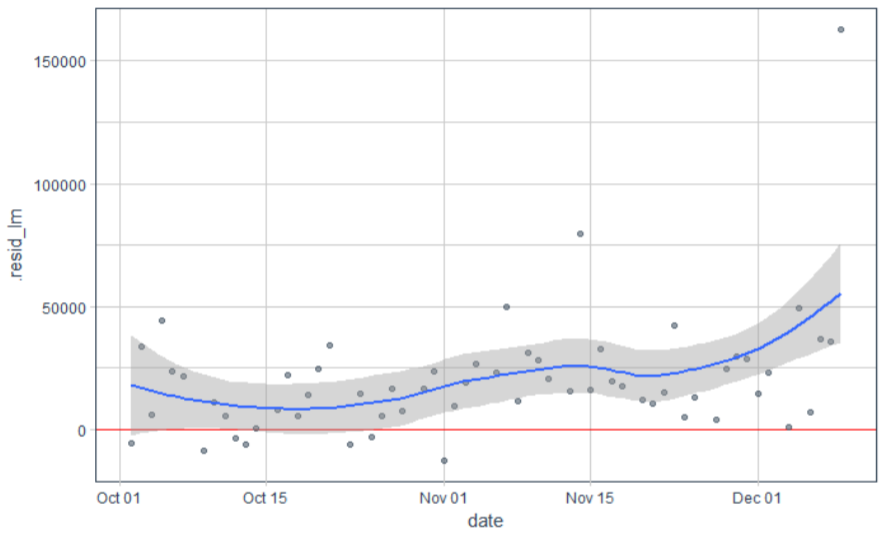


Figure 16: Residual plot of testing data

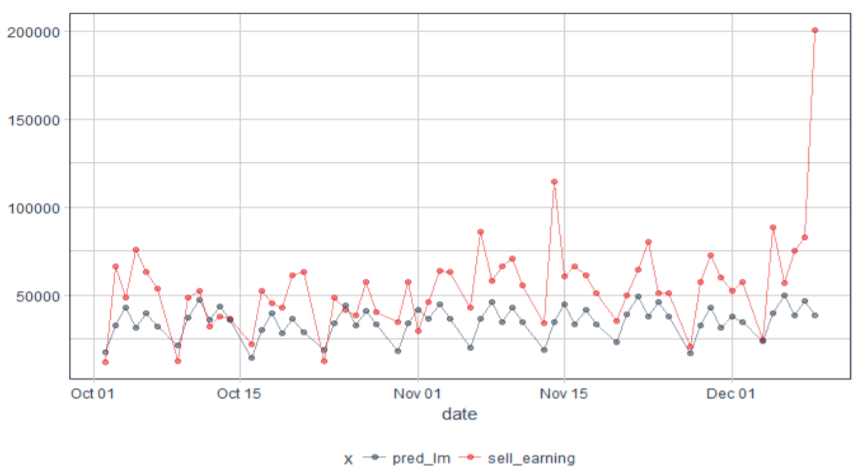


Figure 17: Actual vs predicted earning for test data

We have plotted predicted versus actual sell earning for our test data in figure 17. We have noticed that our model predicted test data in a steady manner.

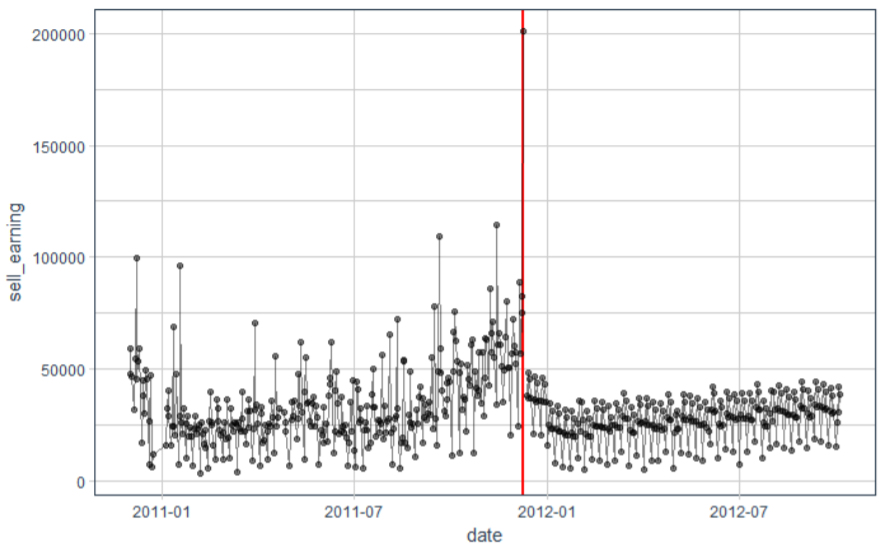


Figure 18: Model's prediction of earning from sell

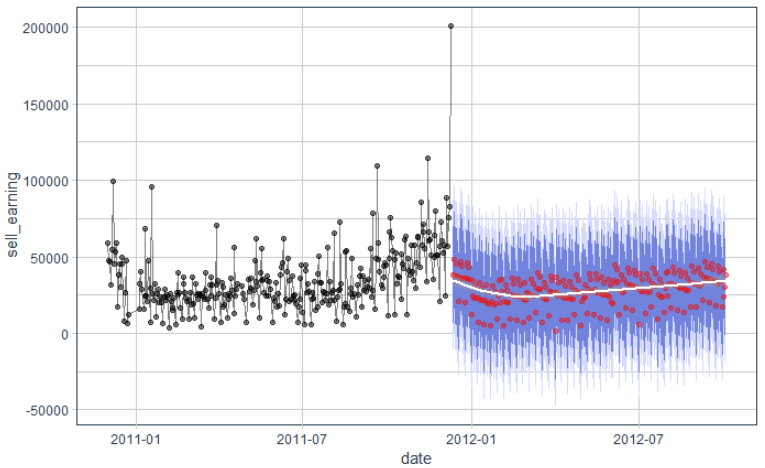


Figure 19: Forecast with a confidence interval

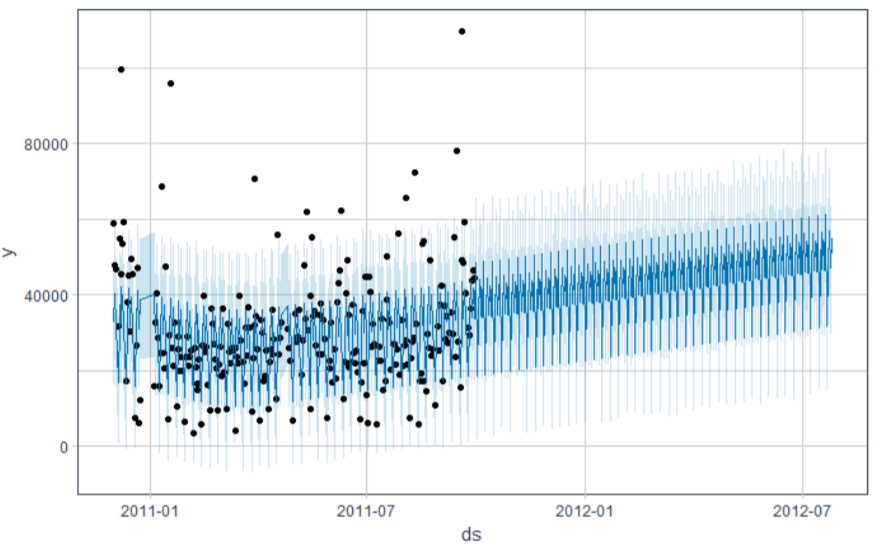


Figure 20: Forecast with Prophet model

# Discussion

As in figure 18, we have seen some deviation in accuracy, we would like to consider the standard deviation of the test residual to modify our model. We have then plotted the forecast showing the confidence of intervals in figure 19. This projection shows us that the average earning might be ranged between 0 to 5000 throughout the year. We have plotted another forecasting using Facebook's Prophet Model [3] shown in figure 20. This projection shows that there might be a higher trend of earning in the future time slot.

# Conclusion

The idea of this project was to learn how to do forecasting using available data. During this project, we have faced difficulties during data processing as we have no Saturdays in our data. Also, there were multiple days off for different occasion throughout the year on which the data were not available. Our model could perform better in forecasting if we could train the model with more data. Also, the trend of rising and fall of earning among seasons cannot be cross-checked due to lack of data.

# Reference

[1] M. Hudson, “The Pros and Cons of E-Commerce and Selling Online”, The Balance, February 15, 2017. [2] Online Retail Data Set, Machine Learning Repository, Available at, https://archive.ics.uci.edu/ml/datasets/Online+Retail   
[3] https://github.com/facebook/prophet

# Appendix

Online\_Retail\_Data\_Analysis

Md\_Fahd\_Al\_Georgy

## R Markdown

#-----------------------------------------Initial Setup----------------------------------------------  
#====================================================================================================  
#create a function to check for installed packages and install them if they are not installed  
install <- function(packages){  
 new.packages <- packages[!(packages %in% installed.packages()[, "Package"])]  
 if (length(new.packages))   
 install.packages(new.packages, dependencies = TRUE)  
 sapply(packages, require, character.only = TRUE)  
}  
  
# usage  
required.packages <- c("caret", "broom", "ggcorrplot","matrixStats", "tidyverse", "timetk", "prophet", "tidyquant", "modelr", "gridExtra", "grid")  
install(required.packages)

## Loading required package: caret

## Warning: package 'caret' was built under R version 3.5.1

## Loading required package: lattice

## Loading required package: ggplot2

## Warning: package 'ggplot2' was built under R version 3.5.1

## Loading required package: broom

## Warning: package 'broom' was built under R version 3.5.1

## Loading required package: ggcorrplot

## Warning: package 'ggcorrplot' was built under R version 3.5.2

## Loading required package: matrixStats

## Warning: package 'matrixStats' was built under R version 3.5.2

## Loading required package: tidyverse

## Warning: package 'tidyverse' was built under R version 3.5.2

## -- Attaching packages ---------------------------------- tidyverse 1.2.1 --

## v tibble 1.4.2 v purrr 0.2.5  
## v tidyr 0.8.1 v dplyr 0.7.6  
## v readr 1.1.1 v stringr 1.3.1  
## v tibble 1.4.2 v forcats 0.4.0

## Warning: package 'tibble' was built under R version 3.5.1

## Warning: package 'tidyr' was built under R version 3.5.1

## Warning: package 'readr' was built under R version 3.5.1

## Warning: package 'purrr' was built under R version 3.5.1

## Warning: package 'dplyr' was built under R version 3.5.1

## Warning: package 'stringr' was built under R version 3.5.1

## Warning: package 'forcats' was built under R version 3.5.2

## -- Conflicts ------------------------------------- tidyverse\_conflicts() --  
## x dplyr::count() masks matrixStats::count()  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()  
## x purrr::lift() masks caret::lift()

## Loading required package: timetk

## Warning: package 'timetk' was built under R version 3.5.2

## Loading required package: prophet

## Warning: package 'prophet' was built under R version 3.5.2

## Loading required package: Rcpp

## Warning: package 'Rcpp' was built under R version 3.5.2

## Loading required package: rlang

## Warning: package 'rlang' was built under R version 3.5.1

##   
## Attaching package: 'rlang'

## The following objects are masked from 'package:purrr':  
##   
## %@%, %||%, as\_function, flatten, flatten\_chr, flatten\_dbl,  
## flatten\_int, flatten\_lgl, invoke, list\_along, modify, prepend,  
## rep\_along, splice

## Loading required package: tidyquant

## Warning: package 'tidyquant' was built under R version 3.5.2

## Loading required package: lubridate

## Warning: package 'lubridate' was built under R version 3.5.1

##   
## Attaching package: 'lubridate'

## The following object is masked from 'package:base':  
##   
## date

## Loading required package: PerformanceAnalytics

## Warning: package 'PerformanceAnalytics' was built under R version 3.5.2

## Loading required package: xts

## Warning: package 'xts' was built under R version 3.5.2

## Loading required package: zoo

## Warning: package 'zoo' was built under R version 3.5.1

##   
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':  
##   
## as.Date, as.Date.numeric

##   
## Attaching package: 'xts'

## The following objects are masked from 'package:dplyr':  
##   
## first, last

##   
## Attaching package: 'PerformanceAnalytics'

## The following object is masked from 'package:graphics':  
##   
## legend

## Loading required package: quantmod

## Warning: package 'quantmod' was built under R version 3.5.2

## Loading required package: TTR

## Warning: package 'TTR' was built under R version 3.5.2

## Version 0.4-0 included new data defaults. See ?getSymbols.

## Loading required package: modelr

## Warning: package 'modelr' was built under R version 3.5.2

##   
## Attaching package: 'modelr'

## The following object is masked from 'package:broom':  
##   
## bootstrap

## Loading required package: gridExtra

## Warning: package 'gridExtra' was built under R version 3.5.1

##   
## Attaching package: 'gridExtra'

## The following object is masked from 'package:dplyr':  
##   
## combine

## Loading required package: grid

## caret broom ggcorrplot matrixStats tidyverse timetk   
## TRUE TRUE TRUE TRUE TRUE TRUE   
## prophet tidyquant modelr gridExtra grid   
## TRUE TRUE TRUE TRUE TRUE

options(na.action = na.warn)

write.csv(retail\_data, "new\_retail\_data.csv")  
str(retail\_data)

## Classes 'tbl\_df', 'tbl' and 'data.frame': 541909 obs. of 14 variables:  
## $ InvoiceNo : chr "536365" "536365" "536365" "536365" ...  
## $ StockCode : chr "85123A" "71053" "84406B" "84029G" ...  
## $ Description: chr "WHITE HANGING HEART T-LIGHT HOLDER" "WHITE METAL LANTERN" "CREAM CUPID HEARTS COAT HANGER" "KNITTED UNION FLAG HOT WATER BOTTLE" ...  
## $ Quantity : int 6 6 8 6 6 2 6 6 6 32 ...  
## $ InvoiceDate: POSIXct, format: "2010-12-01 08:26:00" "2010-12-01 08:26:00" ...  
## $ UnitPrice : num 2.55 3.39 2.75 3.39 3.39 7.65 4.25 1.85 1.85 1.69 ...  
## $ CustomerID : int 17850 17850 17850 17850 17850 17850 17850 17850 17850 13047 ...  
## $ Country : chr "United Kingdom" "United Kingdom" "United Kingdom" "United Kingdom" ...  
## $ sell\_date : Date, format: "2010-12-01" "2010-12-01" ...  
## $ day\_of\_week: Ord.factor w/ 7 levels "Sun"<"Mon"<"Tue"<..: 4 4 4 4 4 4 4 4 4 4 ...  
## $ sell\_time : 'hms' num 08:26:00 08:26:00 08:26:00 08:26:00 ...  
## ..- attr(\*, "units")= chr "secs"  
## $ month : chr "12" "12" "12" "12" ...  
## $ earning : num 15.3 20.3 22 20.3 20.3 ...  
## $ sell\_status: chr "sold" "sold" "sold" "sold" ...

colSums(is.na(retail\_data))

## InvoiceNo StockCode Description Quantity InvoiceDate UnitPrice   
## 0 0 1454 0 0 0   
## CustomerID Country sell\_date day\_of\_week sell\_time month   
## 135080 0 0 0 0 0   
## earning sell\_status   
## 0 0

sapply(retail\_data[,c(2,7,8)],function(x)length(unique(x)))

## StockCode CustomerID Country   
## 4070 4373 38

unique(retail\_data$Country)

## [1] "United Kingdom" "France" "Australia"   
## [4] "Netherlands" "Germany" "Norway"   
## [7] "EIRE" "Switzerland" "Spain"   
## [10] "Poland" "Portugal" "Italy"   
## [13] "Belgium" "Lithuania" "Japan"   
## [16] "Iceland" "Channel Islands" "Denmark"   
## [19] "Cyprus" "Sweden" "Austria"   
## [22] "Israel" "Finland" "Bahrain"   
## [25] "Greece" "Hong Kong" "Singapore"   
## [28] "Lebanon" "United Arab Emirates" "Saudi Arabia"   
## [31] "Czech Republic" "Canada" "Unspecified"   
## [34] "Brazil" "USA" "European Community"   
## [37] "Malta" "RSA"

plot1 <- retail\_data %>%  
 filter(Country == "United Kingdom") %>%  
 ggplot(aes(x = Country, fill = sell\_status)) +  
 geom\_bar(alpha = .6) +  
 scale\_fill\_tq(values = palette\_dark()) +  
 theme\_tq() +  
 theme(axis.text.x = element\_text(angle = 45, vjust = 1, hjust = 1)) +  
 guides(fill = FALSE) +  
 labs(x = "")  
  
plot2 <- retail\_data %>%  
 filter(Country != "United Kingdom") %>%  
 ggplot(aes(x = Country, fill = sell\_status)) +  
 geom\_bar(alpha = .6) +  
 scale\_fill\_tq(values = palette\_green()) +  
 theme\_tq() +  
 theme(legend.position = "right") +  
 theme(axis.text.x = element\_text(angle = 90, vjust = 1, hjust = .9)) +  
 labs(x = "",fill = "")  
title <- textGrob("Country-wise Selling Trend", gp = gpar(fontface = "bold", cex = 1.5))  
grid.arrange(plot1, plot2, top=title ,widths = c(0.2, 1.2))

retail\_data %>%  
 ggplot(aes(x = sell\_date, color = sell\_status)) + ggtitle("Number of sells/returns over time") +   
 theme(plot.title = element\_text(hjust = 0.5))+  
 facet\_grid(sell\_status ~ ., scales = "free") +  
 geom\_freqpoly(bins = 80, size = 1, alpha = 0.9) +  
 scale\_color\_tq(values = palette\_dark())

retail\_data %>%  
 ggplot(aes(x = sell\_date, y = ..density.., color = sell\_status)) + ggtitle("Relationship between sell and return over time") +   
 theme(plot.title = element\_text(hjust = 0.5))+  
 geom\_freqpoly(size = 1, alpha = 0.9, bins = 60) +  
 scale\_color\_manual(values = palette\_light())

retail\_data %>%  
 ggplot(aes(x = sell\_time, y = sell\_date)) + ggtitle("Transactions trend over the time of the day") + theme(plot.title = element\_text(hjust = 0.5))+  
 stat\_bin\_2d(alpha = 1, bins = 19, color = "grey") +  
 scale\_fill\_gradientn(colours = c(palette\_green()[[1]], palette\_dark()[[2]]))

retail\_data %>%  
 mutate(day = format(InvoiceDate, "%d")) %>%  
 group\_by(month, day) %>%  
 summarise(Earnings = sum(earning)) %>%  
 ggplot(aes(x = month, y = day, fill = Earnings)) + ggtitle("Total earnings per day over the months") + theme(plot.title = element\_text(hjust = 0.5))+ ylab("Days of the month")+   
 geom\_tile(alpha = 1, color = "grey") +  
 scale\_fill\_gradientn(colours = c(palette\_green()[[1]], palette\_dark()[[2]])) +  
 theme(legend.position = "right")

retail\_data %>%  
 group\_by(StockCode, Description) %>%  
 summarise(Total\_Quantity = sum(Quantity)) %>%  
 arrange(-Total\_Quantity) %>% head()

## # A tibble: 6 x 3  
## # Groups: StockCode [6]  
## StockCode Description Total\_Quantity  
## <chr> <chr> <int>  
## 1 84077 WORLD WAR 2 GLIDERS ASSTD DESIGNS 53847  
## 2 85099B JUMBO BAG RED RETROSPOT 47363  
## 3 84879 ASSORTED COLOUR BIRD ORNAMENT 36381  
## 4 22197 POPCORN HOLDER 36334  
## 5 21212 PACK OF 72 RETROSPOT CAKE CASES 36039  
## 6 85123A WHITE HANGING HEART T-LIGHT HOLDER 35025

p1 <- retail\_data %>%  
 group\_by(StockCode, Description) %>%  
 summarise(Total\_Quantity = sum(Quantity)) %>%  
 ggplot(aes(x = Total\_Quantity)) +  
 geom\_density(fill = palette\_light()[[1]], alpha = 0.8) +  
 theme\_tq()  
  
p2 <- retail\_data %>%  
 group\_by(StockCode, Description) %>%  
 summarise(Total\_Quantity = sum(Quantity)) %>%  
 filter(Total\_Quantity > 1) %>%  
 ggplot(aes(x = Total\_Quantity)) +  
 geom\_density(fill = palette\_light()[[1]], alpha = 0.8) +  
 theme\_tq()  
  
p3 <- retail\_data %>%  
 group\_by(StockCode, Description) %>%  
 summarise(Total\_Quantity = sum(Quantity)) %>%  
 filter(Total\_Quantity > 10000) %>%  
 ggplot(aes(x = Total\_Quantity)) +  
 geom\_density(fill = palette\_light()[[1]], alpha = 0.8) +  
 theme\_tq()  
title1 <- textGrob("Selling trend of product by quantity", gp = gpar(fontface = "bold", cex = 1.5))   
grid.arrange(p1, p2, p3, ncol = 3,top=title1 ,widths = c(0.6, 0.5, 0.7))

top\_sold\_items <- retail\_data %>%  
 group\_by(sell\_date, StockCode, Description) %>%  
 summarise(sum = sum(Quantity)) %>%  
 group\_by(StockCode, Description)%>%  
 summarise(n = n()) %>%  
 arrange(-n)  
  
top\_items\_transactions <- retail\_data %>%  
 filter(StockCode %in% top\_sold\_items$StockCode[1:5]) %>%  
 group\_by(sell\_date, StockCode) %>%  
 summarise(sum = sum(Quantity)) %>%  
 spread(key = StockCode, value = sum)  
  
retail\_data %>%  
 filter(StockCode %in% top\_sold\_items$StockCode[1:5]) %>%  
 group\_by(sell\_date, StockCode, Description) %>%  
 summarise(sum = sum(Quantity)) %>%  
 ggplot(aes(x = sell\_date, y = sum)) +  
 facet\_wrap(~ StockCode, ncol = 1, scales = "free") +  
 geom\_line(color = palette\_light()[[1]], size = 1, alpha = 0.8) +  
 theme\_tq() +  
 labs(Title= "Top 5 Items' Selling Trends", x = "",   
 y = "Quantity of Sold Items")

retail\_data %>%  
 filter(StockCode == "85123A") %>%  
 group\_by(sell\_date, sell\_status) %>%  
 summarise(Most\_sold\_item = sum(Quantity)) %>%  
 ggplot(aes(x = sell\_date, y = Most\_sold\_item, color = sell\_status)) + ggtitle("Transaction History of Most Sold Item") + theme(plot.title = element\_text(hjust = 0.5))+ ylab("Item Quantities")+  
 facet\_wrap(~ sell\_status, ncol = 1, scales = "free") +  
 geom\_line(size = .4, alpha = 1) +   
 scale\_color\_tq(values = palette\_dark())

multiple\_transactions <- retail\_data %>%  
 group\_by(sell\_date, CustomerID) %>%  
 summarise(sum = sum(Quantity)) %>%  
 group\_by(CustomerID) %>%  
 summarise(n = n()) %>%  
 mutate(revisited\_customers = ifelse(n > 1, "reoccurring\_customers", "one\_time\_customers"))  
  
length(which(multiple\_transactions$revisited\_customers == "reoccurring\_customers"))

## [1] 2992

revisited\_customers\_sell\_date <- left\_join(retail\_data, multiple\_transactions, by = "CustomerID") %>%  
 distinct(sell\_date, CustomerID, revisited\_customers) %>%  
 group\_by(sell\_date, revisited\_customers) %>%  
 summarise(n = n()) %>%  
 spread(key = revisited\_customers, value = n)

## Warning: `chr\_along()` is soft-deprecated as of rlang 0.2.0.  
## This warning is displayed once per session.

multiple\_transactions %>%  
 group\_by(revisited\_customers) %>%  
 summarise(n = n()) %>%  
 mutate(prop = n / sum(n)) %>%  
 ggplot(aes(x = "", y = prop, fill = revisited\_customers)) +ggtitle("Ratio of one-time customers & revisiting customers") + theme(plot.title = element\_text(hjust = 0.5))+ ylab("")+ xlab("")+  
 geom\_bar(stat = "identity", alpha = 1) +  
 coord\_polar("y", start = 0) +  
 scale\_fill\_tq(values = palette\_dark())

purchases <- retail\_data %>%  
 group\_by(sell\_date, CustomerID) %>%  
 summarise(n = n(),  
 Tot\_item = sum(Quantity),  
 Tot\_earning = sum(earning)) %>%  
 group\_by(sell\_date) %>%  
 summarise(mean\_earning\_per\_customer = mean(Tot\_earning),  
 mean\_quantity\_sold\_per\_customer = mean(Tot\_item),  
 mean\_items\_sold\_per\_customer = mean(n))  
purchases %>%  
 gather(x, y, mean\_earning\_per\_customer:mean\_items\_sold\_per\_customer) %>%  
 ggplot(aes(x = sell\_date, y = y)) + ggtitle("Selling Trend per Customer") + theme(plot.title = element\_text(hjust = 0.5))+ ylab("")+ xlab("")+  
 facet\_wrap(~ x, ncol = 1, scales = "free") +  
 geom\_line(color = palette\_light()[[1]], size = 1, alpha = 1) +  
 geom\_smooth(color = palette\_light()[[2]], method = 'loess')

## Warning: `lang()` is soft-deprecated as of rlang 0.2.0.  
## Please use `call2()` instead  
## This warning is displayed once per session.

## Warning: `new\_overscope()` is soft-deprecated as of rlang 0.2.0.  
## Please use `new\_data\_mask()` instead  
## This warning is displayed once per session.

## Warning: `overscope\_eval\_next()` is soft-deprecated as of rlang 0.2.0.  
## Please use `eval\_tidy()` with a data mask instead  
## This warning is displayed once per session.

returned\_items <- retail\_data %>%  
 group\_by(sell\_date, sell\_status) %>%  
 summarise(total\_quantity = sum(Quantity)) %>%  
 spread(key = sell\_status, value = total\_quantity)  
   
returned\_items%>%  
 gather(x, y, sold:returned) %>%  
 ggplot(aes(x = sell\_date, y = y, color = x)) +   
 geom\_line(size = 1, alpha = 0.8) +  
 scale\_color\_manual(values = palette\_light()) +  
 theme\_tq() +  
 labs(x = "",   
 y = "quantity of items",  
 title = "Purchase and Return Over the Time")

item\_transactions <- retail\_data %>%  
 group\_by(sell\_date, StockCode) %>%  
 summarise(n = n()) %>%  
 group\_by(sell\_date) %>%  
 summarise(item\_transactions = n())  
item\_transactions %>%  
 ggplot(aes(x = sell\_date, y = item\_transactions)) +  
 geom\_line(color = palette\_light()[[1]], size = 1, alpha = 0.8) +  
 geom\_smooth(color = palette\_light()[[2]], method = 'loess') +  
 theme\_tq() +  
 labs(title= "Item Transactions Over the Time" ,x = "",   
 y = "number of different items",  
 color = "")

net\_earning <- retail\_data %>%  
 group\_by(sell\_date) %>%  
 summarise(sum\_earning = sum(earning),  
 mean\_earning = mean(earning),  
 sum\_quantity = sum(Quantity),  
 mean\_quantity = mean(Quantity))  
net\_earning %>%  
 gather(x, y, sum\_earning:mean\_quantity) %>%  
 ggplot(aes(x = sell\_date, y = y)) +  
 facet\_wrap(~ x, ncol = 1, scales = "free") +  
 geom\_line(color = palette\_light()[[1]], size = 1, alpha = 0.8) +  
 geom\_smooth(color = palette\_light()[[2]], method = 'loess') +  
 theme\_tq() +  
 labs(title="Mean of Earning and Itme Quantity" ,x = "",   
 y = "")

sold\_items <- retail\_data %>%  
 filter(earning > 0) %>%  
 group\_by(sell\_date) %>%  
 summarise(sell\_earning = sum(earning),  
 #sell\_earning\_mean = mean(earning),  
 sell\_quantity = sum(Quantity))  
 #sell\_quantity\_mean = mean(Quantity))  
sold\_items %>%  
 gather(x, y, sell\_earning:sell\_quantity) %>%  
 ggplot(aes(x = sell\_date, y = y)) +  
 facet\_wrap(~ x, ncol = 1, scales = "free") +  
 geom\_line(color = palette\_light()[[1]], size = 1, alpha = 0.8) +  
 geom\_smooth(color = palette\_light()[[2]], method = 'loess') +  
 theme\_tq() +  
 labs(title= "Sell Trends",x = "",   
 y = "")

return\_items <- retail\_data %>%  
 filter(earning < 0) %>%  
 group\_by(sell\_date) %>%  
 summarise(return\_items\_sum\_price = sum(earning),  
 mean\_return\_items\_price = mean(earning),  
 quantity\_return\_items = sum(Quantity),  
 mean\_quantity\_return\_items = mean(Quantity))  
return\_items %>%  
 gather(x, y, return\_items\_sum\_price:mean\_quantity\_return\_items) %>%  
 ggplot(aes(x = sell\_date, y = y)) +  
 facet\_wrap(~ x, ncol = 1, scales = "free") +  
 geom\_line(color = palette\_light()[[1]], size = 1, alpha = 0.8) +  
 theme\_tq() +  
 labs(x = "",   
 y = "")

unit\_quant <- distinct(select(retail\_data, sell\_date, StockCode, UnitPrice)) %>%  
 mutate(unit\_quant = paste(sell\_date, StockCode, sep = "\_")) %>%  
 select(unit\_quant, UnitPrice)  
  
mean\_unit\_price <- retail\_data %>%  
 filter(sell\_status == "sold") %>%  
 group\_by(sell\_date, StockCode) %>%  
 summarise(n = n()) %>%  
 mutate(unit\_quant = paste(sell\_date, StockCode, sep = "\_")) %>%  
 left\_join(unit\_quant, by = "unit\_quant") %>%  
 group\_by(sell\_date, StockCode) %>%  
 summarise(mean = mean(UnitPrice)) %>%  
 group\_by(sell\_date) %>%  
 summarise(mean\_unit\_price = mean(mean))  
mean\_unit\_price %>%  
 ggplot(aes(x = sell\_date, y = mean\_unit\_price)) +  
 geom\_line(color = palette\_light()[[1]], size = 1, alpha = 0.8) +  
 theme\_tq() +  
 labs(x = "",   
 y = "mean unit price of sold items")

# 

#Forecasting

transaction\_per\_day <- distinct(select(retail\_data, sell\_date, day\_of\_week, month)) %>%  
 left\_join(net\_earning, by = "sell\_date") %>%  
 left\_join(mean\_unit\_price, by = "sell\_date") %>%  
 left\_join(sold\_items, by = "sell\_date") %>%  
 left\_join(return\_items, by = "sell\_date") %>%  
 left\_join(purchases, by = "sell\_date") %>%  
 left\_join(revisited\_customers\_sell\_date, by = "sell\_date") %>%  
 left\_join(returned\_items, by = "sell\_date") %>%  
 left\_join(item\_transactions, by = "sell\_date") %>%  
 left\_join(top\_items\_transactions, by = "sell\_date") %>%  
 mutate(diff\_sum\_earning = sell\_earning - lag(sell\_earning),  
 season = ifelse(month %in% c("03", "04", "05"), "spring",  
 ifelse(month %in% c("06", "07", "08"), "summer",  
 ifelse(month %in% c("09", "10", "11"), "fall", "winter"))))

transaction\_per\_day <- transaction\_per\_day %>%  
 mutate(model = ifelse(sell\_date <= "2011-10-01", "train", "test"))  
  
colnames(transaction\_per\_day)[grep("^[0-9]+", colnames(transaction\_per\_day))] <- paste0("P\_", colnames(transaction\_per\_day)[grep("^[0-9]+", colnames(transaction\_per\_day))])

transaction\_per\_day %>%  
 ggplot(aes(x = sell\_date, y = sell\_earning, color = model)) +  
 geom\_point(alpha = 0.5) +  
 geom\_line(alpha = 0.5) +  
 scale\_color\_manual(values = palette\_light()) +  
 theme\_tq()

augmented\_transactions <- transaction\_per\_day %>%  
 rename(date = sell\_date) %>%  
 select(model, date, sell\_earning) %>%   
 tk\_augment\_timeseries\_signature() %>%  
 select(-contains("month"))

## Warning: `list\_len()` is soft-deprecated as of rlang 0.2.0.  
## Please use `new\_list()` instead  
## This warning is displayed once per session.

augmented\_transactions <- augmented\_transactions[complete.cases(augmented\_transactions), ]

(var <- data.frame(colnames = colnames(augmented\_transactions[, sapply(augmented\_transactions, is.numeric)]),  
 colvars = colVars(as.matrix(augmented\_transactions[, sapply(augmented\_transactions, is.numeric)]))) %>%  
 filter(colvars == 0))

## colnames colvars  
## 1 hour 0  
## 2 minute 0  
## 3 second 0  
## 4 hour12 0  
## 5 am.pm 0

augmented\_transactions <- select(augmented\_transactions, -one\_of(as.character(var$colnames)))

# 

# Removing highly correlated features

relation <- cor(augmented\_transactions[, sapply(augmented\_transactions, is.numeric)])  
p.relation <- cor\_pmat(augmented\_transactions[, sapply(augmented\_transactions, is.numeric)])  
  
ggcorrplot(relation, type = "upper", outline.col = "white", hc.order = TRUE, p.mat = p.relation,  
 colors = c(palette\_light()[1], "white", palette\_light()[2]))

correlation <- findCorrelation(relation, cutoff=0.8)   
augmented\_transactions <- select(augmented\_transactions, -one\_of(colnames(relation)[correlation]))  
train <- filter(augmented\_transactions, model == "train") %>%  
 select(-model)  
test <- filter(augmented\_transactions, model == "test")

fit\_lm <- glm(sell\_earning ~ ., data = train)  
tidy(fit\_lm) %>%  
 gather(x, y, estimate:p.value) %>%  
 ggplot(aes(x = term, y = y, color = x, fill = x)) +  
 facet\_wrap(~ x, scales = "free", ncol = 2) +  
 geom\_bar(stat = "identity", alpha = 0.8) +  
 scale\_color\_manual(values = palette\_light()) +  
 scale\_fill\_manual(values = palette\_light()) +  
 theme\_tq() + labs(x="", y="")+  
 theme(axis.text.x = element\_text(angle = 75, vjust = 1, hjust = 1))

augment(fit\_lm) %>%  
 ggplot(aes(x = date, y = .resid)) +  
 geom\_hline(yintercept = 0, color = "red") +  
 geom\_point(alpha = 0.5, color = palette\_light()[[1]]) +  
 geom\_smooth() +  
 theme\_tq()

## `geom\_smooth()` using method = 'loess' and formula 'y ~ x'

# Visualizing prediction test

pred\_test <- test %>%  
 add\_predictions(fit\_lm, "pred\_lm") %>%  
 add\_residuals(fit\_lm, ".resid\_lm")

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type =  
## ifelse(type == : prediction from a rank-deficient fit may be misleading  
  
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type =  
## ifelse(type == : prediction from a rank-deficient fit may be misleading

pred\_test %>%  
 ggplot(aes(x = date, y = .resid\_lm)) +  
 geom\_hline(yintercept = 0, color = "red") +  
 geom\_point(alpha = 0.5, color = palette\_light()[[1]]) +  
 geom\_smooth() +  
 theme\_tq()

## `geom\_smooth()` using method = 'loess' and formula 'y ~ x'

pred\_test %>%  
 gather(x, y, sell\_earning, pred\_lm) %>%  
 ggplot(aes(x = date, y = y, color = x)) +  
 geom\_point(alpha = 0.5) +  
 geom\_line(alpha = 0.5) +  
 scale\_color\_manual(values = palette\_light()) +  
 theme\_tq()

# Forecasting

# Extract index  
time\_index <- transaction\_per\_day %>%  
 tk\_index()

augmented\_transactions %>%  
 ggplot(aes(x = date, y = diff)) +  
 geom\_point(alpha = 0.5, aes(color = as.factor(diff))) +  
 geom\_line(alpha = 0.5) +  
 scale\_color\_manual(values = palette\_light()) +  
 theme\_tq()

augmented\_transactions %>%  
 select(date, wday.lbl, diff) %>%  
 filter(wday.lbl != "Sunday" & diff > 86400) %>%  
 mutate(days\_missing = diff / 86400 -1)

## # A tibble: 5 x 4  
## date wday.lbl diff days\_missing  
## <date> <ord> <int> <dbl>  
## 1 2011-01-04 Tuesday 1036800 11  
## 2 2011-04-26 Tuesday 432000 4  
## 3 2011-05-03 Tuesday 172800 1  
## 4 2011-05-31 Tuesday 172800 1  
## 5 2011-08-30 Tuesday 172800 1

off\_days <- c("2010-12-24", "2010-12-25", "2010-12-26", "2010-12-27", "2010-12-28", "2010-12-29", "2010-12-30", "2010-01-01", "2010-01-02", "2010-01-03",  
 "2011-04-22", "2011-04-23", "2011-04-24", "2011-04-25", "2011-05-02", "2011-05-30", "2011-08-29", "2011-04-29", "2011-04-30") %>%   
 ymd()

future\_time\_index <- time\_index %>%  
 tk\_make\_future\_timeseries(n\_future = 300, inspect\_weekdays = TRUE, inspect\_months = FALSE, skip\_values = off\_days)

## Warning: Creating a condition with `cnd\_signal()` is soft-deprecated as of rlang 0.3.0.  
## Please use `signal()` instead.  
## This warning is displayed once per session.

## Warning: `.mufflable` is soft-deprecated as of rlang 0.3.0 and no longer has any effect  
## This warning is displayed once per session.

## pad applied on the interval: day

## The following `skip\_values` were not in the future date sequence: 2010-12-24, 2010-12-25, 2010-12-26, 2010-12-27, 2010-12-28, 2010-12-29, 2010-12-30, 2010-01-01, 2010-01-02, 2010-01-03, 2011-04-22, 2011-04-23, 2011-04-24, 2011-04-25, 2011-05-02, 2011-05-30, 2011-08-29, 2011-04-29, 2011-04-30

future\_time\_index %>%  
 tk\_get\_timeseries\_signature() %>%  
 ggplot(aes(x = index, y = diff)) +  
 geom\_point(alpha = 0.5, aes(color = as.factor(diff))) +  
 geom\_line(alpha = 0.5) +  
 scale\_color\_manual(values = palette\_light()) +  
 theme\_tq()

## Warning: Removed 1 rows containing missing values (geom\_point).

## Warning: Removed 1 rows containing missing values (geom\_path).

future\_data <- future\_time\_index %>%  
 tk\_get\_timeseries\_signature() %>%  
 rename(date = index)  
  
prediction <- predict(fit\_lm, newdata = future\_data)

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type =  
## ifelse(type == : prediction from a rank-deficient fit may be misleading

prediction <- future\_data %>%  
 select(date) %>%  
 add\_column(sell\_earning = prediction)  
transaction\_per\_day %>%  
 select(sell\_date, sell\_earning) %>%  
 rename(date = sell\_date) %>%  
 rbind(prediction) %>%  
 ggplot(aes(x = date, y = sell\_earning)) +  
 scale\_x\_date() +  
 geom\_vline(xintercept = as.numeric(max(transaction\_per\_day$sell\_date)), color = "red", size = 1) +  
 geom\_point(alpha = 0.5) +  
 geom\_line(alpha = 0.5) +  
 theme\_tq()

## Warning: Removed 1 rows containing missing values (geom\_point).

residual\_testing <- pred\_test$.resid\_lm  
sd\_of\_res\_testing <- sd(residual\_testing, na.rm = TRUE)  
  
prediction <- prediction %>%  
 mutate(  
 lo.95 = sell\_earning - 1.96 \* sd\_of\_res\_testing,  
 lo.80 = sell\_earning - 1.28 \* sd\_of\_res\_testing,  
 hi.80 = sell\_earning + 1.28 \* sd\_of\_res\_testing,  
 hi.95 = sell\_earning + 1.96 \* sd\_of\_res\_testing  
 )

transaction\_per\_day %>%  
 select(sell\_date, sell\_earning) %>%  
 rename(date = sell\_date) %>%  
 ggplot(aes(x = date, y = sell\_earning)) +  
 geom\_point(alpha = 0.5) +  
 geom\_line(alpha = 0.5) +  
 geom\_ribbon(aes(ymin = lo.95, ymax = hi.95), data = prediction,   
 fill = "#D5DBFF", color = NA, size = 0) +  
 geom\_ribbon(aes(ymin = lo.80, ymax = hi.80, fill = key), data = prediction,  
 fill = "#596DD5", color = NA, size = 0, alpha = 0.8) +  
 geom\_point(aes(x = date, y = sell\_earning), data = prediction,  
 alpha = 0.5, color = palette\_light()[[2]]) +  
 geom\_smooth(aes(x = date, y = sell\_earning), data = prediction,  
 method = 'loess', color = "white") +  
 theme\_tq()

## Warning: Removed 1 rows containing non-finite values (stat\_smooth).

## Warning: Removed 1 rows containing missing values (geom\_point).

# 

transaction\_per\_day <- transaction\_per\_day %>%  
 mutate(model = ifelse(sell\_date <= "2011-10-01", "train2", "test2"))  
  
train2 <- filter(transaction\_per\_day, model == "train2") %>%  
 select(sell\_date, sell\_earning) %>%  
 rename(ds = sell\_date,  
 y = sell\_earning)  
  
test2 <- filter(transaction\_per\_day, model == "test2") %>%  
 select(sell\_date, sell\_earning) %>%  
 rename(ds = sell\_date)

closed <- data.frame(ds = as.Date(c("2010-12-24", "2010-12-25", "2010-12-26", "2010-12-27", "2010-12-28",   
 "2010-12-29", "2010-12-30", "2010-01-01", "2010-01-02", "2010-01-03",  
 "2011-04-22", "2011-04-23", "2011-04-24", "2011-04-25", "2011-05-02",   
 "2011-05-30", "2011-08-29", "2011-04-29", "2011-04-30"))) %>%  
 mutate(holiday = paste0("off\_day\_", seq\_along(1:length(ds))))  
prophet\_model\_test <- prophet(train2,   
 growth = "linear", # growth curve trend  
 n.changepoints = 100, # Prophet automatically detects changes in trends by selecting changepoints from the data  
 yearly.seasonality = FALSE, # yearly seasonal component using Fourier series  
 weekly.seasonality = TRUE, # weekly seasonal component using dummy variables  
 holidays = closed)

## Disabling daily seasonality. Run prophet with daily.seasonality=TRUE to override this.

## Warning: `env\_bind\_fns()` is soft-deprecated as of rlang 0.3.0.  
## Please use `env\_bind\_active()` instead.  
## This warning is displayed once per session.

## Warning: `overscope\_clean()` is soft-deprecated as of rlang 0.2.0.  
## This warning is displayed once per session.

forecasting <- predict(prophet\_model\_test, test2)

## Warning: Unknown or uninitialised column: 'y'.

forecasting %>%  
 mutate(.resid2 = trend - yhat) %>%  
 ggplot(aes(x = ds, y = .resid2)) +  
 geom\_hline(yintercept = 0, color = "red") +  
 geom\_point(alpha = 0.5, color = palette\_light()[[1]]) +  
 geom\_smooth() +  
 theme\_tq()

## `geom\_smooth()` using method = 'loess' and formula 'y ~ x'

future <- make\_future\_dataframe(prophet\_model\_test, periods = 300)  
forecast <- predict(prophet\_model\_test, future)  
plot(prophet\_model\_test, forecast) +  
 theme\_tq()

1. https://www.sisense.com/glossary/retail-analytics/ [↑](#footnote-ref-1)
2. https://www.sisense.com/glossary/retail-analytics/ [↑](#footnote-ref-2)
3. https://www.nchannel.com/blog/how-to-perform-sales-trend-analysis/ [↑](#footnote-ref-3)
4. https://www.vendhq.com/blog/how-retailers-can-use-data-to-boost-productivity-customer-service-sales/ [↑](#footnote-ref-4)
5. https://www.alteryx.com/shopper-insights-insider-perspectives-infographic [↑](#footnote-ref-5)
6. ## https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1470-6431.2008.00707.x (Assessing the demand for value‐based organic meats in Canada: a combined retail and household scanner‐data approach, published on August 13, 2008; by Sven Anders and Anke Moeser)

   [↑](#footnote-ref-6)
7. App Store Mining and Analysis: MSR for App Stores Mark Harman, Yue Jia, and Yuanyuan Zhang University College London, Malet Place, London, WC1E 6BT, UK. (<http://www0.cs.ucl.ac.uk/staff/F.Sarro/projects/UCLappA/resources/MSR12.pdf>) [↑](#footnote-ref-7)
8. National Retail Data Monitor for Public Health Surveillance Michael M. Wagner,1 F-C. Tsui,1 J. Espino,1 W. Hogan,1 J. Hutman,1 J. Hersh,2 D. Neill,3 A. Moore,1,3 G. Parks,1 C. Lewis,4 R. Aller5 1University of Pittsburgh, Pittsburgh, Pennsylvania; 2Pennsylvania Department of Health, Harrisburg, Pennsylvania; 3Carnegie Mellon University, Pittsburgh, Pennsylvania; 4Massachusetts Department of Public Health, Boston, Massachusetts; 5Los Angeles Department of Health, Los Angeles, California [↑](#footnote-ref-8)
9. https://archive.ics.uci.edu/ml/datasets/online+retail [↑](#footnote-ref-9)