Walmart Project: Use market basket analysis to classify shopping trips

Aishwarya Kadam, Shrey Gupta, Tapan Golakiya

Introduction

With the advancement of ecommerce, it is challenging for retail stores like Walmart to maintain or increase their sales, as these days people prefer to get the products with just one touch on their smart phones. For retail sales to increase and to provide better services to their customers, it becomes necessary for the retail stores to find and understand the pattern of the goods being purchased by the customers based on the past customer experience and the goods being bought in the past. For an instance, Milk, eggs, bread etc. are the products which are consumed on daily basis, so, whenever we enter any retail store like Walmart, we find that these products are kept far away from the entrance, so that whenever a customer wants to purchase these goods, they must walk through all the sections to get these products. The strategy behind this idea is that, until the customer reaches the end section, on their way he/she can put the items they see and feel important in their carts and eventually end up buying more products than what they had originally intended to buy. To come up with such strategies, we perform the market basket analysis on the historical data which contained the details about the transactions like, different products in an itemset belonging to their specific departments being purchased on a specific day of the week. Based on this analysis conclusions are made, which help in decision-making for the retailers to stock the products, discontinue or reduce the number of products which consumers do not prefer to buy or increase the quantity of those products which people often buy. Market basket analysis helps in associating the goods that go together by finding the support, confidence and lift of these respective products. With the help of these parameters, the association rules can be formed, which helps the retail stores like Walmart to make the shopping experience better for the customers thereby gaining profit by increasing their sales. In this project we use this analysis to classify the trips of the customers. These trips are classified into 38 distinct categories.

Dataset

We have used the dataset from Kaggle: <https://www.kaggle.com/c/walmart-recruiting-trip-type-classification>. There were two datasets available for the analysis, training set and test dataset. Training data had 647054 observations and 7 attributes, in which one of the attributes was the labelled output data, ‘TripType’, which was used to classify the shopping trips. Test data had 653646 observations and 6 attributes.

The 7 attributes are,

* Visit Number- An Id corresponding to a single trip by a single customer, basically a transaction Id.
* Weekday-The day of the week on which the trip was made to the store.
* UPC- The UPC number of the product purchased, which is a unique Id of the product.
* ScanCount - The number of the given item that was purchased. A negative value indicated that the product was returned.
* DepartmentDescription- A high-level description of the department to which the item belonged. There are total 69 distinct Departments.
* FinelineNumber - A more refined category for each of the products, created by Walmart.
* TripType - A categorical id representing the type of shopping trip the customer made.

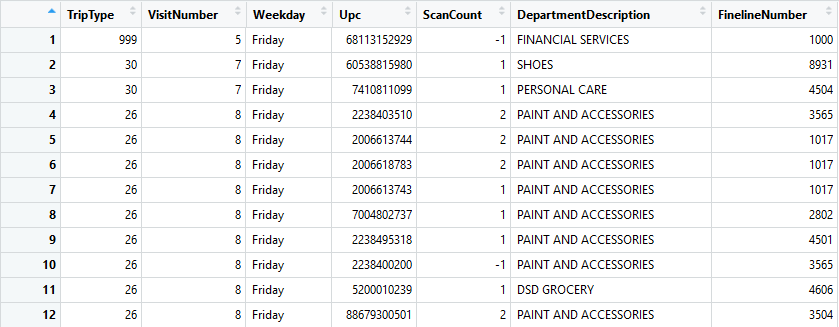


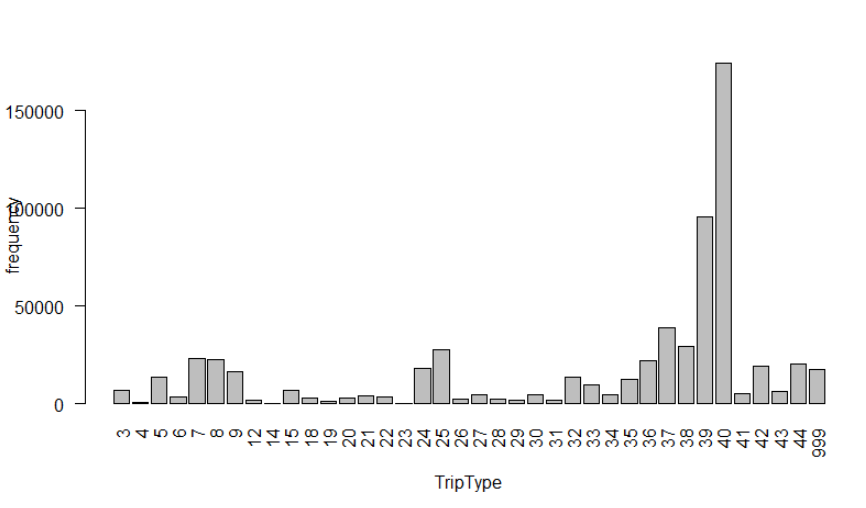
Figure 1: This figure shows the first 12 sample observations of our dataset

Data Visualization

1. We have plotted a histogram of the frequency/count of each trip type.

#R code for plotting frequency of each Trip Type

barplot(table(train\_data$TripType), las=2,xlab='TripType', ylab='frequency')

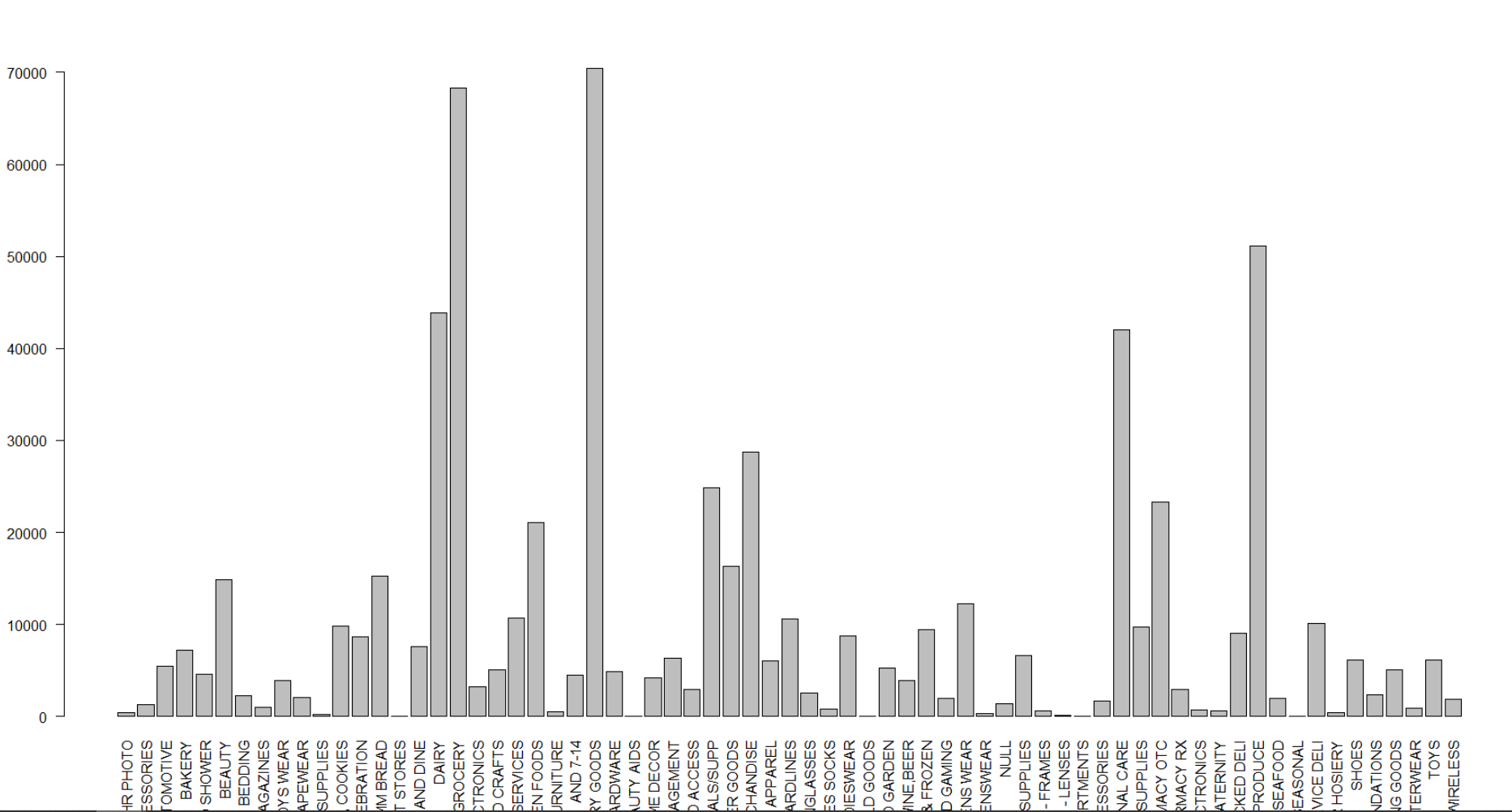


We see from the above plot that maximum number of people made a trip of type 40 and very few people made a trip of type 4 or 14 or 23.

1. We have plotted a histogram of the frequency/count of each DepartmentDescription.

# R code for plotting frequency of each Department Description

barplot(table(train\_data$DepartmentDescription), las=2,xlab='DepartmentDescription', ylab='frequency')



We observe that, "GROCERY" and "GROCERY AND DRY GOODS" have the highest frequency counts.

1. We have plotted a pie chart of the frequency/count of each day of the week.

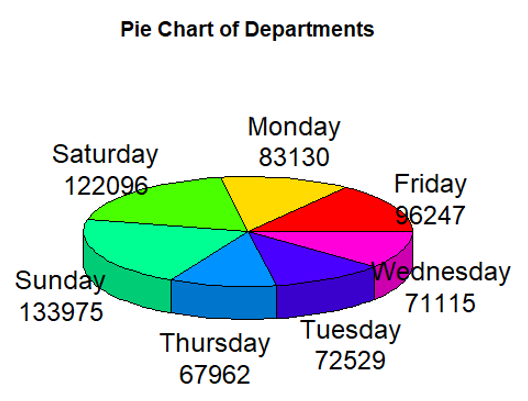
#R code for plotting the frequency/count of transactions made on each day of week.

library(plotrix)

table\_weekday <- table(train\_data$Weekday)

labels\_weekday <- paste(names(table\_weekday), "\n", table\_weekday, sep="")

pie3D(table\_weekday, labels = labels\_weekday, main="Pie Chart of Departments")

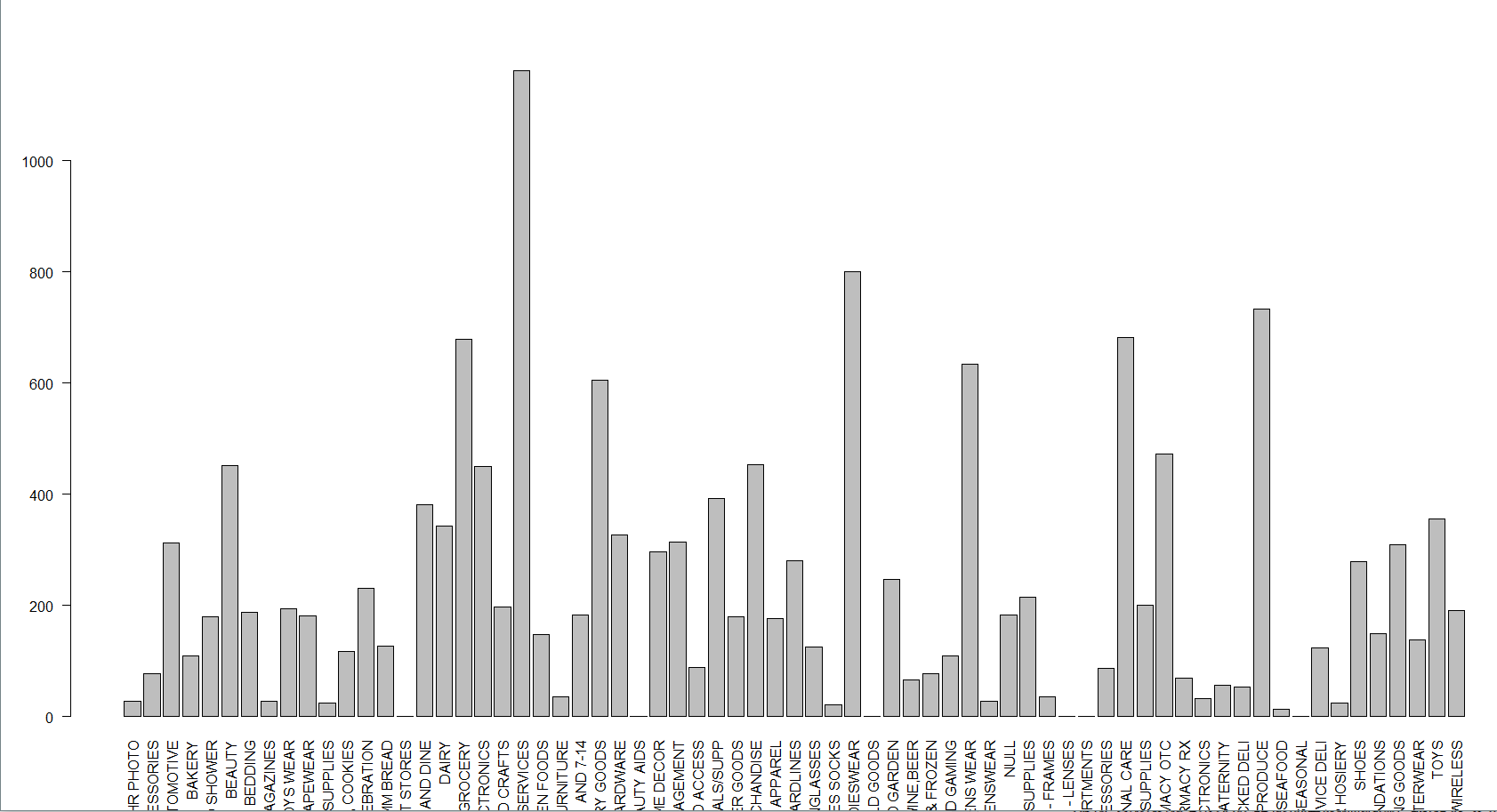


From the above pie chart, we observe that most of the transactions were made on weekends, So, we can assume that most people prefer to do shopping when they have luxury time to spend.

1. We have plotted a bar plot of the frequency/count of returned items of each department.

#R code for plotting the frequency/count of returned items of each department.

barplot(table(train\_data$DepartmentDescription), las=2)

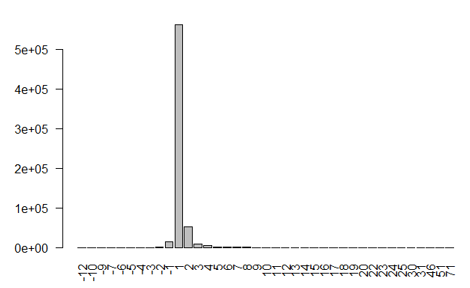


We observe that, products from the department "FINANCIAL SERVICES" seem to be returned most number of times at the store by customers.

1. We have plotted a bar plot of the frequency/count of ScanCount feature. This means the number of times an item has been scanned.

#R code for plotting frequency of each ScanCount of the item

barplot(table(train\_data$ScanCount), las=2)

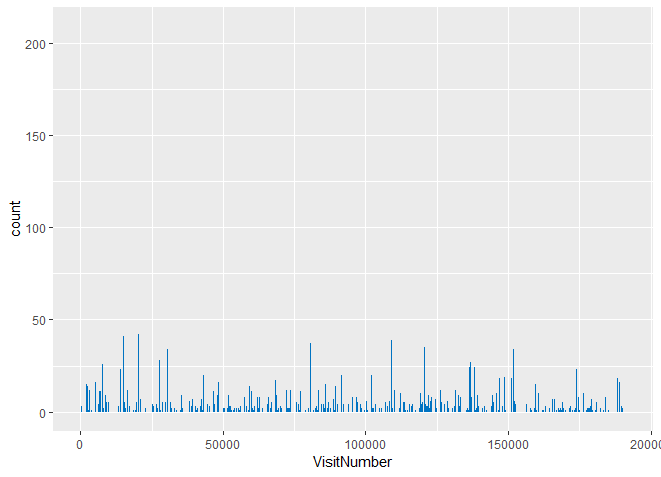


We observe that most number of products were scanned just once.

1. We have plotted a bar plot of the frequency/count of items in each transaction (VisitNumber)..

#R code for plotting the frequency/count of items in each transaction (VisitNumber).

ggplot(train\_data, aes(VisitNumber)) + geom\_bar(fill = "#0073C2FF")



Data Pre-processing

For cleaning the data, we removed all the rows which had missing values for the UPC number and its corresponding Fineline number, as it makes no sense to run market basket analysis on a dataset which does not have any information about the product id of the product being purchased. There were total 4129 observations in which the product id was not specified. So, after removing these samples, the training data reduced to 642925 samples and 7 attributes

#Rcode for finding missing values

sum(is.na(train\_data$FinelineNumber))

row.has.na <- apply(train\_data, 1, function(x){any(is.na(x))})

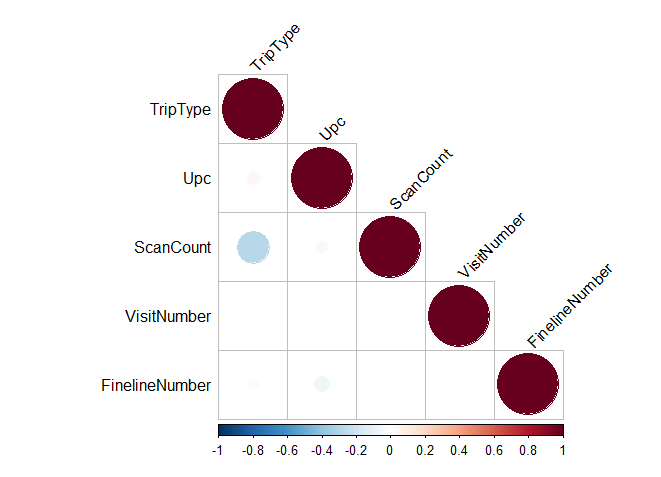
sum(row.has.na)

train\_data <- train\_data[!row.has.na,]

rownames(train\_data) <- seq(length=nrow(train\_data))

summary(train\_data)

We applied correlation filter to our dataset to check if any correlation existed between the features of our dataset. However, our dataset did not have any correlation among the features which can be seen in the figure below.

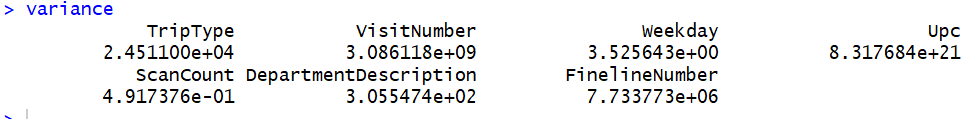


We applied the variance filter to remove the features which had very less variance and we observed that ScanCount had a variance of 0.49 which is very low when compared to other attributes

#Rcode for variance filter

variance <- sapply(train\_data, var)

train\_data <- train\_data[, -c(5)]



We created a validation set by dividing the training dataset in 80:20 ratio by taking random samples from the dataset to check the accuracy of the training.

The new training data then reduced to 514341 training samples and 7 attributes and the new validation data would be 128584 validation samples and 7 attributes.

The test data has 653646 observations and 6 attributes.

#R code for splitting the data

library(caTools)

set.seed(123)

split = sample.split(train\_data$TripType, SplitRatio = 0.80)

training\_set2 = subset(train\_data, split == TRUE)

test\_set1 = subset(train\_data, split == FALSE)

test\_set2 <- test\_set1[,2:6]

rownames(training\_set2) <- seq(length=nrow(training\_set2))

rownames(test\_set1) <- seq(length=nrow(test\_set1))

rownames(test\_set2) <- seq(length=nrow(test\_set2))

Implementation

We applied Apriori algorithm in the beginning of our analysis to form the strong association rules from the data which represents the two items that are bought together. We set the minimum support to be 0.0006 and confidence threshold to be 0.5

Various classifiers were implemented to do the analysis on the dataset. Initially we applied Naïve Bayes classifier to the training and validation dataset. The accuracy we achieved was 32.96% and the log loss was about 5.013. Since the accuracy was less than 50%, we have used this classifier as the baseline for other models.

#R Code for Naïve bayes classifier

library(e1071)

classifier\_nb = naiveBayes(TripType~., data = training\_set2)

pred\_nb = predict(classifier\_nb,test\_set2)

cm\_nb = table(test\_set1[,1], pred\_nb)

cm\_nb

#install.packages('caret')

library(caret)

confusionMatrix(cm\_nb)

RMSE <- mean((as.numeric(test\_set1[,1])-as.numeric(pred\_nb))^2)

log(RMSE)

Next, we applied the gradient boosting algorithm XgBoost classifier to classify the trip types. XGBoost, short for (extreme) gradient boosting, is a fast, portable and distributed implementation of the gradient boosting (trees) algorithm.

Gradient boosting is an ensemble (i.e. meta) machine learning algorithm that builds a strong model based on many weaker ones sequentially.

The accuracy we achieved was 51.69% and the log loss was about 1.379 for the 300th iteration. The parameters that we selected for the classifier are as follows:  
max\_depth = 12, nthread = 8, nrounds = 300, num\_class = 38, eta = 0.1, early\_stopping\_rounds = 10, min\_child\_weight = 3. The accuracy we achieved was 51.69% and log loss was about 1.379