SalesForecastingAttempt3

Ghazalpreet Kaur

November 25, 2018

In order to perform day-level analysis, the data must be cleaned and aggregated by date, across all stores.

First we load our data. The datasets that we use are Train, Test and Store.

train <- read\_csv("train.csv", col\_types = list(StateHoliday = col\_character()))  
  
test <- read\_csv("test.csv", col\_types = list(StateHoliday = col\_character()))  
  
store <- read\_csv("store.csv")

## Parsed with column specification:  
## cols(  
## Store = col\_integer(),  
## StoreType = col\_character(),  
## Assortment = col\_character(),  
## CompetitionDistance = col\_integer(),  
## CompetitionOpenSinceMonth = col\_integer(),  
## CompetitionOpenSinceYear = col\_integer(),  
## Promo2 = col\_integer(),  
## Promo2SinceWeek = col\_integer(),  
## Promo2SinceYear = col\_integer(),  
## PromoInterval = col\_character()  
## )

dim(train)

## [1] 1017209 9

dim(test)

## [1] 41088 8

dim(store)

## [1] 1115 10

names(train)

## [1] "Store" "DayOfWeek" "Date" "Sales"   
## [5] "Customers" "Open" "Promo" "StateHoliday"   
## [9] "SchoolHoliday"

names(store)

## [1] "Store" "StoreType"   
## [3] "Assortment" "CompetitionDistance"   
## [5] "CompetitionOpenSinceMonth" "CompetitionOpenSinceYear"   
## [7] "Promo2" "Promo2SinceWeek"   
## [9] "Promo2SinceYear" "PromoInterval"

names(test)

## [1] "Id" "Store" "DayOfWeek" "Date"   
## [5] "Open" "Promo" "StateHoliday" "SchoolHoliday"

Getting some more information from the data.

trainst <- merge(train, store, by = "Store")  
  
dim(trainst)

## [1] 1017209 18

Top 10 stores that had the highest sales:

toptenstores = trainst %>%  
 group\_by(trainst$Store, trainst$StoreType) %>%  
 summarize(Total.Sales = sum(Sales), numberCustomer = sum(Customers), AvgSales = mean(Sales), CompDist = mean(CompetitionDistance)) %>%  
 mutate(AvgSalesPerCustomer = Total.Sales/numberCustomer) %>%  
 arrange(desc(Total.Sales))  
  
head(toptenstores, n=10)

## # A tibble: 10 x 7  
## # Groups: trainst$Store [10]  
## `trainst$Store` `trainst$StoreT~ Total.Sales numberCustomer AvgSales  
## <int> <chr> <int> <int> <dbl>  
## 1 262 b 19516842 3204694 20719.  
## 2 817 a 17057867 2454370 18108.  
## 3 562 b 16927322 2924960 17970.  
## 4 1114 a 16202585 2509542 17200.  
## 5 251 a 14896870 1908934 15814.  
## 6 513 a 14252406 1643527 15130.  
## 7 788 a 14082141 1346835 14949.  
## 8 733 b 14067158 3206058 14933.  
## 9 383 a 13489879 1720249 14320.  
## 10 756 a 12911782 1827980 13707.  
## # ... with 2 more variables: CompDist <dbl>, AvgSalesPerCustomer <dbl>

Stores that had the lowest sales:

tail(toptenstores, n=10)

## # A tibble: 10 x 7  
## # Groups: trainst$Store [10]  
## `trainst$Store` `trainst$StoreT~ Total.Sales numberCustomer AvgSales  
## <int> <chr> <int> <int> <dbl>  
## 1 972 a 2402627 272238 2551.  
## 2 186 a 2353548 237019 3105.  
## 3 254 d 2341661 201507 2486.  
## 4 879 d 2340576 216037 3088.  
## 5 841 a 2318635 321495 2461.  
## 6 263 a 2306075 221342 3042.  
## 7 208 c 2302052 324162 2444.  
## 8 198 a 2268273 264690 2408.  
## 9 543 c 2179287 187583 2313.  
## 10 307 a 2114322 240704 2245.  
## # ... with 2 more variables: CompDist <dbl>, AvgSalesPerCustomer <dbl>

The data is unique by day and store level sales observation. Now we combine the test and training data and clean fields:

## [1] 1017209 11

## [1] 41088 11

## [1] "Store" "DayOfWeek" "Date" "Sales" "Customers"  
## [6] "Open" "Promo" "StateHol" "SchoolHol" "IsTest"   
## [11] "Id"

## [1] 1058297 12

Aggregating the data by Date

#We need the data by date. So, for every date, we calculate the stores that were there for that date, whether the information was in the test dataset or not, the average number of stores open on a date, the average stateholiday and school holiday. We also need to have the mean of promo, sales and customers and that will be possible only when the stores are open.   
  
traintestday <- left\_join(summarize(group\_by(traintest, Date),  
 Stores = length(unique(Store)),  
 IsTest = mean(IsTest),  
 AvgOpen = mean(Open, na.rm = TRUE),  
 AvgStateHol = mean(IsStateHol),  
 StateHol = max(StateHol),  
 AvgSchoolHol = mean(SchoolHol)),  
 summarize(group\_by(traintest[traintest$Open == 1,],Date),  
 AvgPromo = mean(Promo),  
 AvgSales = mean(Sales),  
 AvgCust = mean(Customers))  
 , "Date")  
  
  
View(traintestday)  
  
dim(traintestday)

## [1] 990 10

traintestday$Date = as.Date(traintestday$Date, origin = "1970-01-01")  
names(traintestday)

## [1] "Date" "Stores" "IsTest" "AvgOpen"   
## [5] "AvgStateHol" "StateHol" "AvgSchoolHol" "AvgPromo"   
## [9] "AvgSales" "AvgCust"

dim(traintestday)

## [1] 990 10

Adding more day-level fields

traintestday$AvgSalesperCustomer <- traintestday$AvgSales / traintestday$AvgCust  
  
traintestday$Year <- year(traintestday$Date)  
traintestday$Month <- month(traintestday$Date, TRUE) #True for giving the abbreviation of the month and not just the number of the month in the year.  
traintestday$WDay <- wday(traintestday$Date, TRUE)  
View(traintestday$WDay)  
  
traintestday$Week <- strftime(traintestday$Date, format = "%W")  
traintestday$Week <- reorder(traintestday$Week, -as.numeric(traintestday$Week))  
  
traintestday$YearDay <- yday(traintestday$Date) #to get the day of the year  
  
traintestday$MajorStateHol <- ifelse(traintestday$AvgStateHol > 0.25, 1, 0)  
  
traintestday$MajorSchoolHol <- ifelse(traintestday$AvgSchoolHol > 0.25, 1, 0)  
  
View(traintestday$Week)  
  
  
dim(traintestday)

## [1] 990 18

names(traintestday)

## [1] "Date" "Stores" "IsTest"   
## [4] "AvgOpen" "AvgStateHol" "StateHol"   
## [7] "AvgSchoolHol" "AvgPromo" "AvgSales"   
## [10] "AvgCust" "AvgSalesperCustomer" "Year"   
## [13] "Month" "WDay" "Week"   
## [16] "YearDay" "MajorStateHol" "MajorSchoolHol"

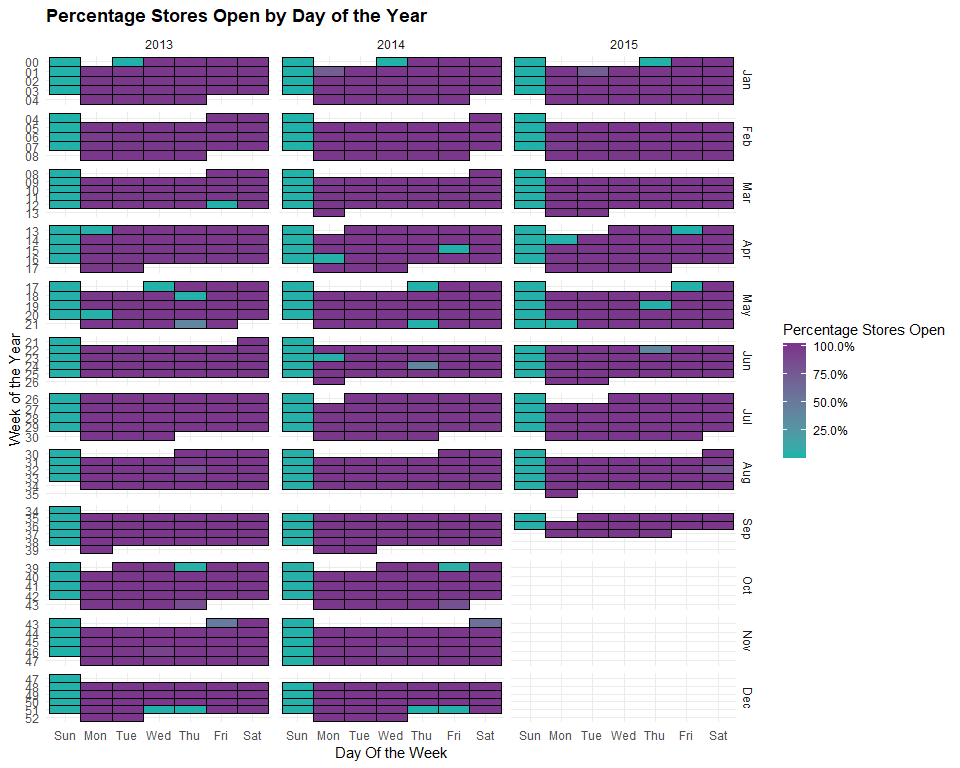
The “traintestday” data frame has all the data describing the day level data. That is, it tells us about the sales, holidays, store related data on a day-level basis.

To find out day-level patterns, we need a calendar visual. This function will help us understand the behavious of all variables throughout the year.

calendar\_fn <- function(var, var\_desc = var, label = comma){  
 traintestday$var <- traintestday[[var]]  
 plot<- ggplot(traintestday, aes(x= WDay, y = Week))+  
 #plot <- ggplot(traintestday, mapping(aes(x=WDay, y=Week)))+  
 facet\_grid(Month ~ Year, scales = "free")+  
 theme\_minimal()+  
 geom\_tile(aes(fill = var ), col = "black")+  
 scale\_fill\_gradient(name = var\_desc,  
 labels = label,  
 low = "lightseagreen",  
 high = "mediumorchid4") +  
 scale\_x\_discrete("Day Of the Week")+  
 scale\_y\_discrete("Week of the Year")+  
 ggtitle(paste(var\_desc, "by Day of the Year"))+  
 theme(plot.title = element\_text(lineheight = .8, face = "bold"))  
 plot  
}

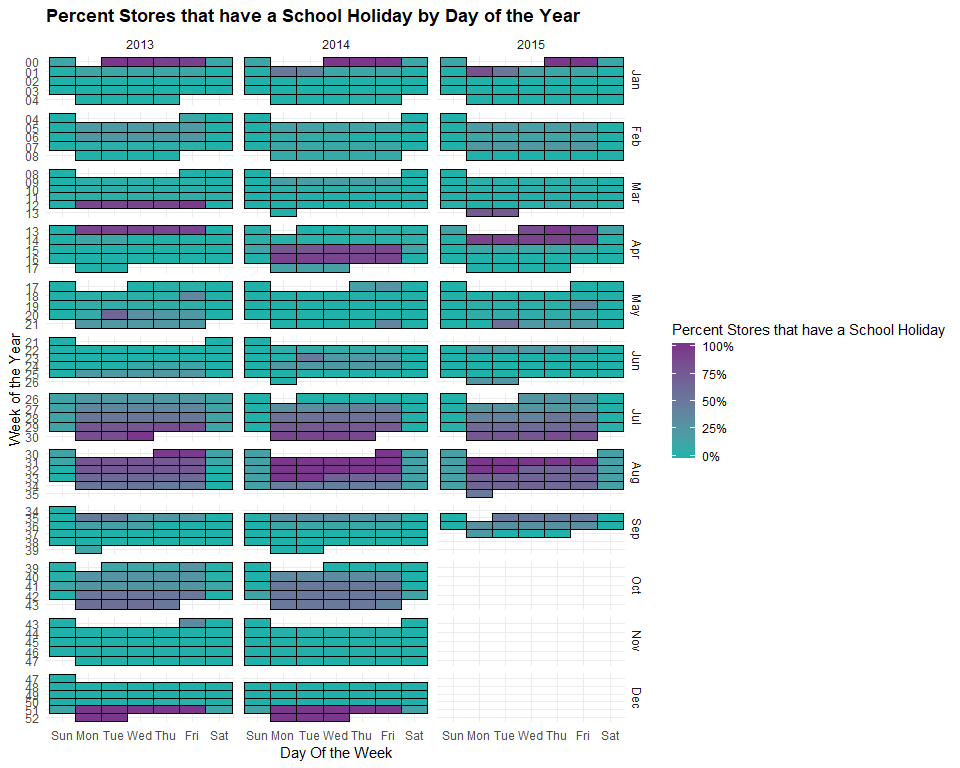
To see for every observation, the percentage of stores that are open

calendar\_fn("AvgOpen", "Percentage Stores Open", percent)

 Almost 100% stores are open most days of the year. Mostly the stores are closed on Sundays and state holidays. The stores that are open on State Holidays, there must be a special reason as to why they are open when others are closed.

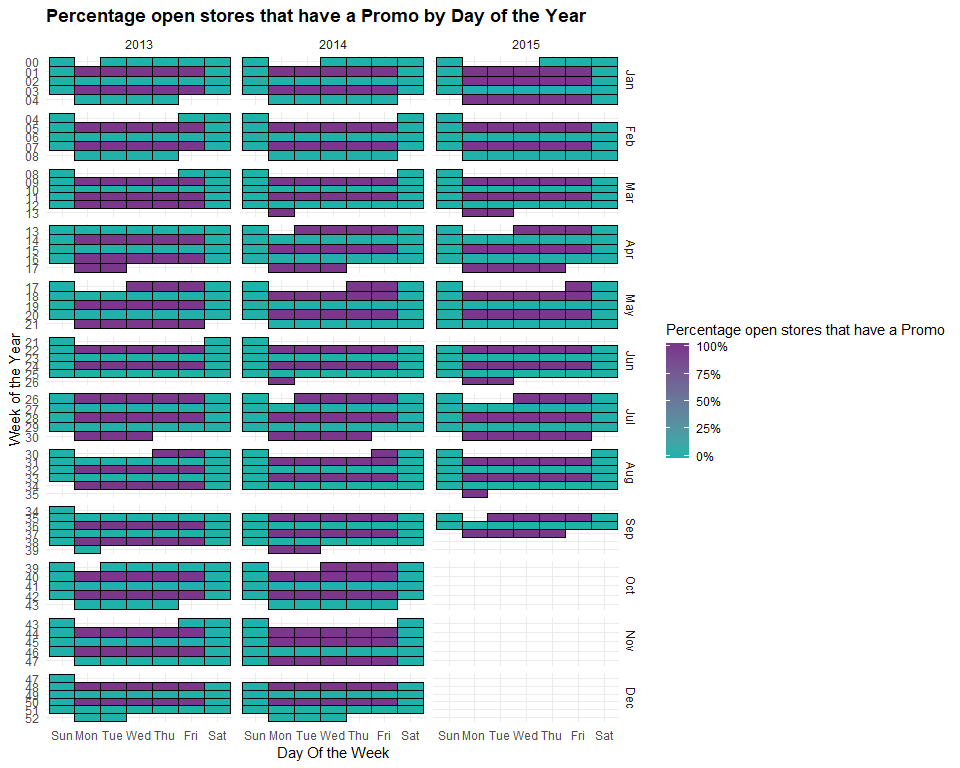
To see the percentage of schools that have a School Holiday

calendar\_fn("AvgSchoolHol", "Percent Stores that have a School Holiday", percent)

 We can clearly see that there is a seasonal pattern here. Most of the stores recognize school holidays between Late July and Early August, New Year’s week and the week prior. Some of the stores recognoze school holidays within October. We also see that the timing of the school holidays shifts from last week of March and first week of April in 2013 to two weeks in April in 2014 and then back to last week of March and First week of April in 2015.

The percentage of stores that have a promotion:

calendar\_fn("AvgPromo", "Percentage open stores that have a Promo", percent)

 Evident from the plot above, there are only two situations possibles. THe store either has a promotion or doesnt. There was no variation in stores that had the promotion or those that did not. Every store had or did not have the promotion during a day.

Taking promotions into consideration, we can see a pattern. There is a promotion every alternate week. Also, we can see some back-to-back promotions. For example: March 2013, April - May 2013, April - May 2014, July - August 2014, September - October 2014, November 2014, January 2015, January - February 2015, and April - May 2015.

Comparing with the real life scenarios, it is quite possible that the second week of these back to back sales has less sales because the promotion has been extended too long and has reached a saturation point.

traintestday$PromotionSaturation <- "None"  
traintestday$PromotionSaturation[traintestday$AvgPromo==1 & lead(traintestday$AvgPromo, 7)==1] <- "Lead"  
traintestday$PromotionSaturation[traintestday$AvgPromo==1 & lag(traintestday$AvgPromo, 7)==1] <- "Lag"  
  
View(traintestday$PromotionSaturation)  
  
dim(traintestday)

## [1] 990 19

We shall now attempt to study Yearly trends in the data. If we see some pattern in the trends, they could be due to some economic factors or other macro factors. Understanding them will help us in future predictions.

Because we are predicting the sales and how and what factors affect it. Sales will be our response variable or the dependant variable. We can try to find the relation between customers and average sales per customer.

#couting number of dates per store  
  
store <- left\_join(store, summarize(group\_by(traintest[traintest$IsTest == 0,], Store), TrainingDates = length(unique(Date))), "Store")  
  
View(store)  
  
dim(store)

## [1] 1115 11

#summarize same store data on a monthly basis  
  
traintest <- left\_join(traintest, store[, c("Store", "TrainingDates")], "Store")  
  
dim(traintest)

## [1] 1058297 13

traintest$MonthYear <- ymd(paste(year(traintest$Date), month(traintest$Date), 1, sep = "-"))  
  
dim(traintest)

## [1] 1058297 14

samestoremonthly <- summarize(group\_by(traintest[traintest$Open == 1 & traintest$TrainingDates != 758, ]  
 , MonthYear)  
 , Observations = length(Date)  
 , AvgSales = mean(Sales)  
 , AvgCust = mean(Customers))  
class(samestoremonthly$MonthYear)

## [1] "Date"

samestoremonthly$MonthYear <- as.Date(samestoremonthly$MonthYear)  
  
samestoremonthly$AverageSalesPerCustomer <- samestoremonthly$AvgSales / samestoremonthly$AvgCust  
  
dim(samestoremonthly)

## [1] 34 5

names(samestoremonthly)

## [1] "MonthYear" "Observations"   
## [3] "AvgSales" "AvgCust"   
## [5] "AverageSalesPerCustomer"

Plotting monthly customer averages:

#{r fig.width= 15, fig.height=6}  
?stat\_smooth

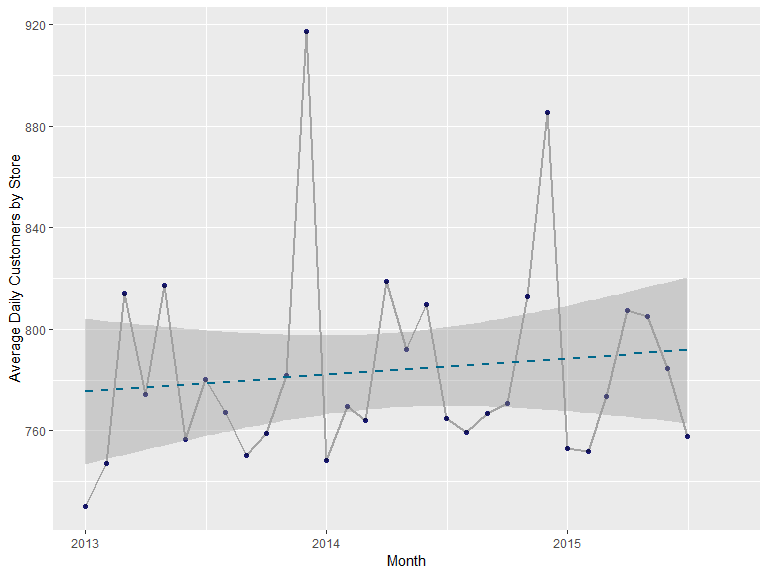
## starting httpd help server ... done

#looking at the linear fit  
  
ggplot(samestoremonthly, aes(MonthYear, AvgCust))+  
 geom\_point(color = "midnightblue" )+  
 geom\_line(alpha = 0.3, size = 1, color = "Black")+  
 stat\_smooth(method = "lm", color = "deepskyblue4", se= TRUE, lty = 2 )+  
 scale\_y\_continuous("Average Daily Customers by Store")+  
 scale\_x\_date("Month")

## Warning: Removed 3 rows containing non-finite values (stat\_smooth).

## Warning: Removed 3 rows containing missing values (geom\_point).

## Warning: Removed 3 rows containing missing values (geom\_path).

 We can clearly see a surge in customers during the holiday season, i.e. December. Each year, March, April and May also perform well each year.

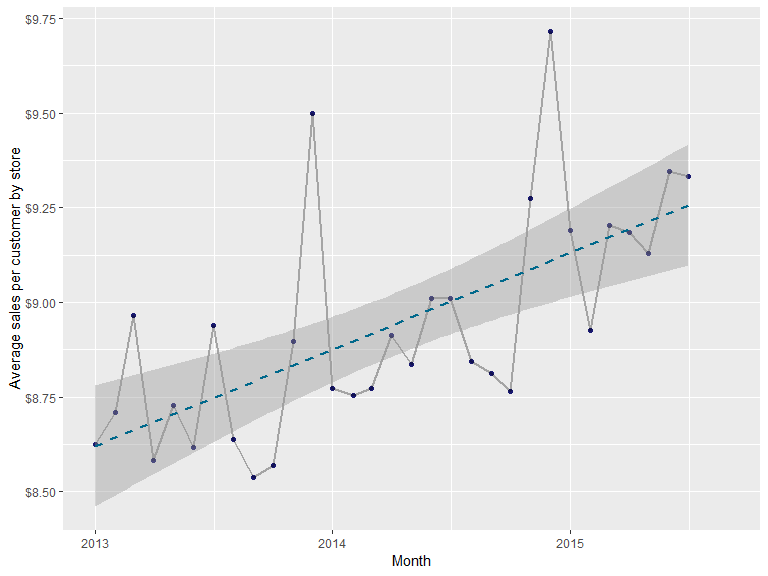
plotting Average sales per customer

ggplot(samestoremonthly, aes(MonthYear, AverageSalesPerCustomer))+  
 geom\_point(color = "midnightblue")+  
 geom\_line(alpha = 0.3 , size = 1, color = "Black")+  
 stat\_smooth(method = "lm", color = "deepskyblue4", se = TRUE, lty = 2)+  
 scale\_y\_continuous("Average sales per customer by store", label = dollar)+  
 scale\_x\_date("Month")

## Warning: Removed 3 rows containing non-finite values (stat\_smooth).

## Warning: Removed 3 rows containing missing values (geom\_point).

## Warning: Removed 3 rows containing missing values (geom\_path).



It turns out that December is the month with the highest average sales per customer. But, unlike the previous plot, March, April and May dont seem to have the same effect on average sales per customer. The trend line seems steeper that the previous plot.

Within-Year Trends

In order to continue studying variation, we can study further in to the day patterns across years and taking into consideration the holidays both state and school and promotion patterns.

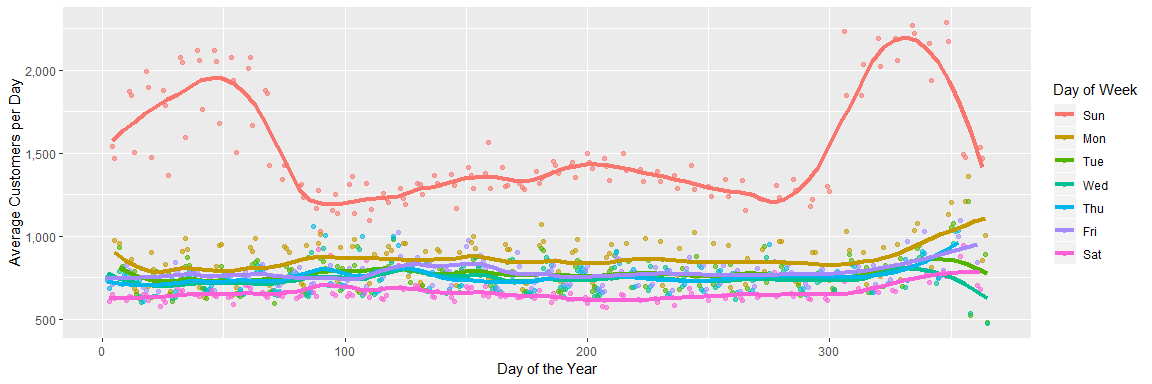
Getting the daily customer trends over the course of a year

ggplot(traintestday[traintestday$AvgStateHol == 0, ]  
 , aes(YearDay, AvgCust, col = WDay))+  
 geom\_point(alpha = .6 )+  
 stat\_smooth(se = FALSE, size = 1.5, span = 0.2)+  
 scale\_x\_continuous("Day of the Year")+  
 scale\_y\_continuous("Average Customers per Day", label = comma)+  
 scale\_color\_discrete("Day of Week")

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

## Warning: Removed 47 rows containing non-finite values (stat\_smooth).

## Warning: Removed 47 rows containing missing values (geom\_point).

 Sundays peak out the most in the number of customers. While Sundays, in general top in the number of customers among other days, the initial Sundays into the year and the Sundays to the end of the year peak in the number

We must remember that on Sundays, most of the stores are closed. Also, from the plot we can see that the relation ship between the days of the year and average customers is not linear, like for the other days. So, we can say that for the very few stores that are open on sundays, have a unique pattern of customers going on

The rest of the days are consistent throughtout the year. Mondays are the most popular among the rest of the days and Saturdays are the least popular. Also to be noted is that towards the end of the year, in December, there is a slight upward trend in the average number of customers except on Tuesdays and Wednesdays.

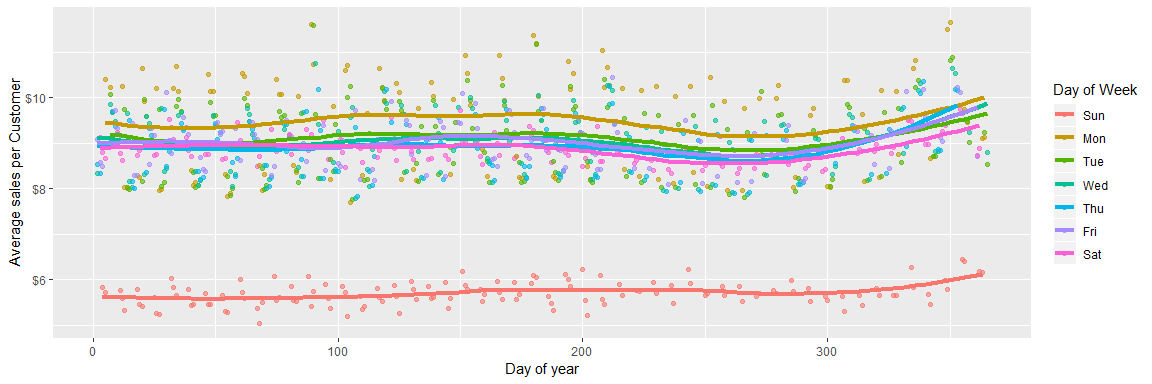
Plotting Average sales per customer every day of the week.

ggplot(traintestday[traintestday$AvgStateHol == 0,]  
 , aes(YearDay, AvgSalesperCustomer, col = WDay))+  
 geom\_point(alpha = 0.6)+  
 stat\_smooth(se = FALSE, size = 1.5, span = 0.5)+  
 scale\_x\_continuous("Day of year")+  
 scale\_y\_continuous("Average sales per Customer", label = dollar)+  
 scale\_color\_discrete("Day of Week")

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

## Warning: Removed 47 rows containing non-finite values (stat\_smooth).

## Warning: Removed 47 rows containing missing values (geom\_point).

 Sundays record substantially smaller sales per customer.

Monday has the highest average sales per customer and Saturday has the lowest.

We can see a lot of variation in average sales per customer in the days.

Similar to the previous plot, we see an upward trend at the end of the year.

Studying State Holidays.

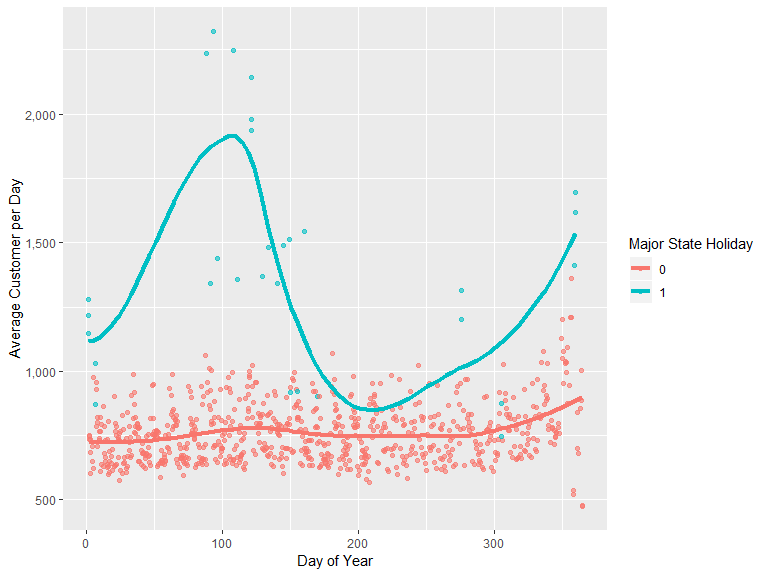
We put a condition previously, where we recognized major state holidays where more that 25% of the stores recognized that state holiday. We can study the customer trends during these major state holidays:

ggplot(traintestday[traintestday$WDay != "Sun",],  
 aes(YearDay, AvgCust, col= as.factor(MajorStateHol)))+  
 geom\_point(alpha = 0.6)+  
 stat\_smooth(se = FALSE, size = 1.5, span = 0.5)+  
 scale\_x\_continuous("Day of Year")+  
 scale\_y\_continuous("Average Customer per Day", label = comma)+  
 scale\_color\_discrete("Major State Holiday")

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

## Warning: Removed 41 rows containing non-finite values (stat\_smooth).

## Warning: Removed 41 rows containing missing values (geom\_point).



We can say that state holidays are the outliers. There also are state holidays that have more average daily customers. We also see that there are days which are not state holidays, but still have high average daily customers.

We can do the same for School Holidays.

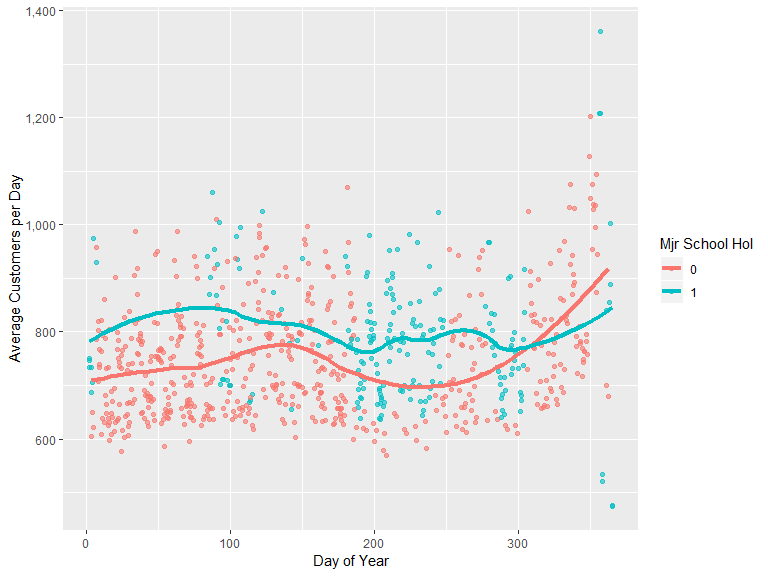
Major School Holidays are the ones where more than 25% stores recognize the school holidays. Studying them as below:

ggplot(traintestday[traintestday$WDay != "Sun" & traintestday$MajorStateHol == 0, ],  
 aes(YearDay, AvgCust, col = as.factor(MajorSchoolHol)))+  
 geom\_point(alpha = 0.6)+  
 stat\_smooth(se = FALSE, size = 1.5, span = 0.5)+  
 scale\_x\_continuous("Day of Year")+  
 scale\_y\_continuous("Average Customers per Day", label = comma)+  
 scale\_color\_discrete("Mjr School Hol")

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

## Warning: Removed 41 rows containing non-finite values (stat\_smooth).

## Warning: Removed 41 rows containing missing values (geom\_point).

 We can see that School holidays have more customers than non-school holidays. Also, the majority school days are around Day 200, which is Late July and early August. We see that similar to the other plots, we can see the similar pattern of heightened sales and number of customers around March, April, December and January.

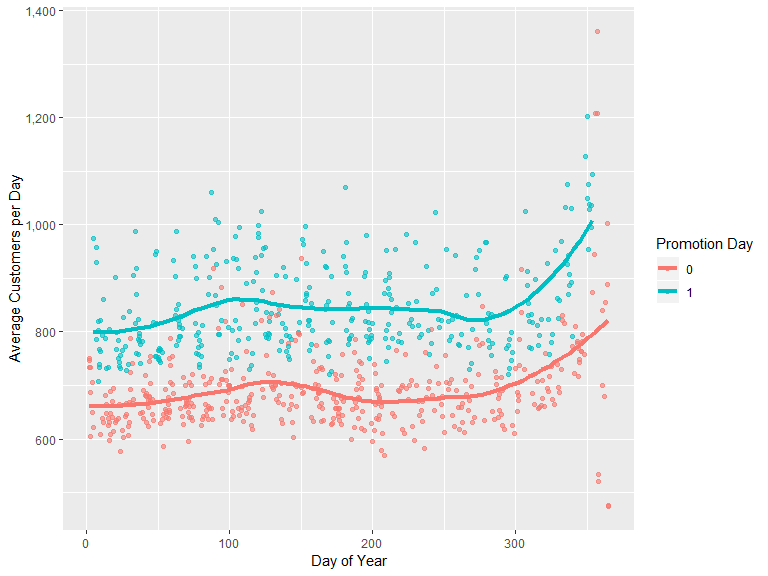
We need to exclude holidays, bith state and school and also Sundays to study the promotions better.

ggplot(traintestday[traintestday$WDay != "Sun" & traintestday$MajorStateHol == 0, ],   
 aes(YearDay, AvgCust, col = as.factor(AvgPromo)))+  
 geom\_point(alpha = 0.6)+  
 stat\_smooth(se= FALSE, size = 1.5, span = 0.5)+  
 scale\_x\_continuous("Day of Year")+  
 scale\_y\_continuous("Average Customers per Day", label = comma)+  
 scale\_color\_discrete("Promotion Day")

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

## Warning: Removed 41 rows containing non-finite values (stat\_smooth).

## Warning: Removed 41 rows containing missing values (geom\_point).



Promotion days definitely have more number of customers that non-promotion days. We can see that, on average, the promotion days have 100 to 150 additional customers per day.

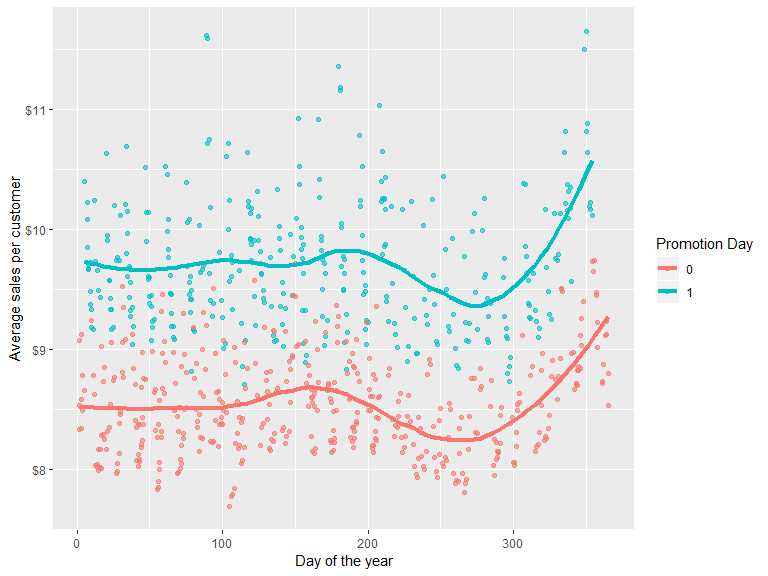
Average Sales per customer:

ggplot(traintestday[traintestday$WDay != "Sun" & traintestday$MajorStateHol == 0, ],  
 aes(YearDay, AvgSalesperCustomer, col = as.factor(AvgPromo)))+  
 geom\_point(alpha = 0.6)+  
 stat\_smooth(se = FALSE, size = 1.5, span = 0.5)+  
 scale\_x\_continuous("Day of the year")+  
 scale\_y\_continuous("Average sales per customer", label = dollar)+  
 scale\_color\_discrete("Promotion Day")

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

## Warning: Removed 41 rows containing non-finite values (stat\_smooth).

## Warning: Removed 41 rows containing missing values (geom\_point).

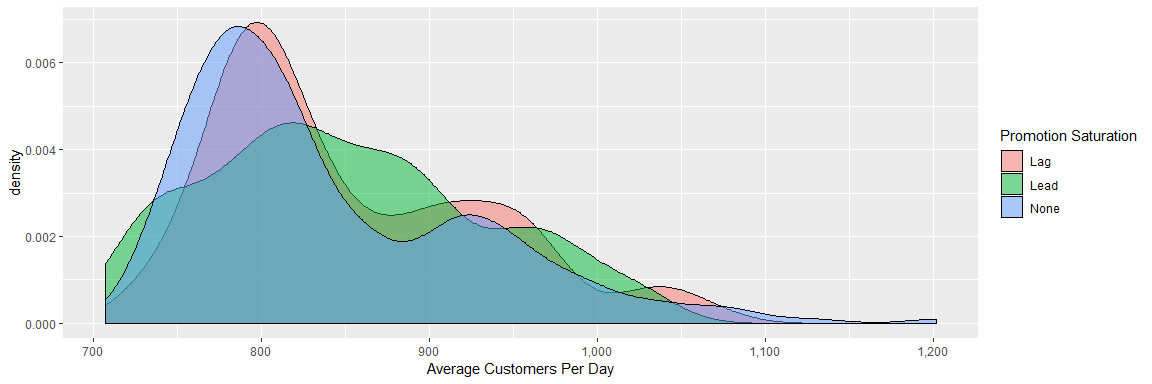
 We see that promotions dont just get more customers but more average sales per customer!. Also, the amount of sales is also high!

Now taking into account the promotions that were back-to-back for 2 weeks. We suspected that the sales might slow down due to promotion saturation. We can confirm our suspicions now!

We have mentioned Lead as the first week of back-to-back promotion and Lag as the second week. Below, None would correspond to the promotiosn that were there only for a week.

ggplot(traintestday[traintestday$WDay != "Sun" & traintestday$MajorStateHol == 0 & traintestday$AvgPromo ==1, ], aes(AvgCust, fill = PromotionSaturation))+  
 geom\_density(adjust = 0.7, alpha = 0.5)+  
 scale\_x\_continuous("Average Customers Per Day", label = comma)+  
 scale\_fill\_discrete("Promotion Saturation")

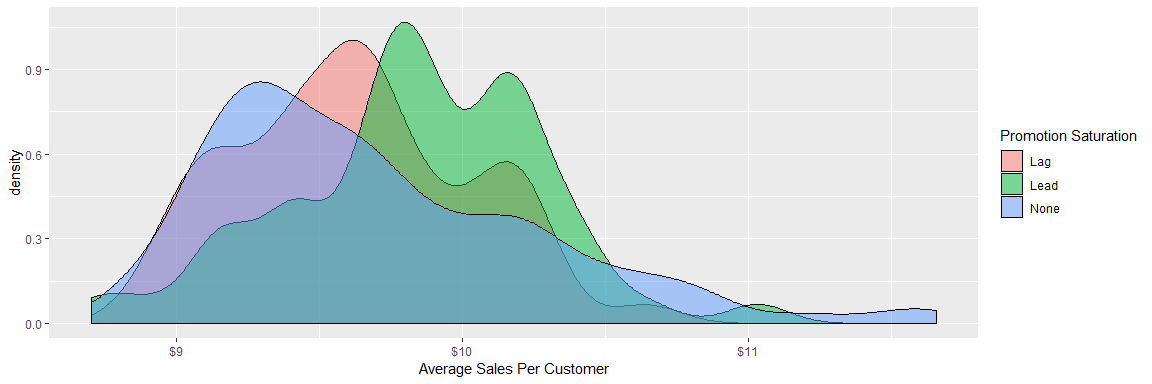
## Warning: Removed 19 rows containing non-finite values (stat\_density).

 We can see, from the above plot that Lag week and the None week, have very similar pattern. Whereas, the Lead week manages to get more average customers per day!

let us check the same for sales.

ggplot(traintestday[traintestday$WDay != "Sun" & traintestday$MajorStateHol == 0 & traintestday$AvgPromo == 1, ], aes(AvgSalesperCustomer, fill = PromotionSaturation))+  
 geom\_density(adjust = 0.7 , alpha = 0.5)+  
 scale\_x\_continuous("Average Sales Per Customer", label = dollar)+  
 scale\_fill\_discrete("Promotion Saturation")

## Warning: Removed 19 rows containing non-finite values (stat\_density).

 We can clearly see that average sales are more for Lead and Lag than for None.

We can witness a business strategy here. We see that back-to-back promotions averages more customers and higher sales than non-promotion weeks. It is only logical to hold these promotions frequently after taking into consideration the related costs.

What does our analysis suggest?

From the raw data that we had in the form of csv files and tables, we have been able to succesfully learn the below points:

\*\* Increasing Average sales per customer is a trend that has been constant throughout the years.

\*\* Promotions have a quite significant positive effect on sales and also on the number of customers that come to the store.

\*\* Having back-to-back sales has proven to be beneficial and the saturation effect that we suspected is not that strong.

\*\* There are a few stores that are open on Sundays. For the ones that are open, the sales per customer and the number of customers have a very unique trend, which appear to be non-linear as opposed to other days.

\*\* State Holidays also have a good positive effect on the store sales and the number of customers.