Predicting Weekly Sales at WalMart Stores

August 31, 2015

1. Executive Summary

Retail stores need to be able to predict sales forecasts for the future and study the effect how strategic offers affect sales, especially during holiday season. Since the number of days in holidays are limited, it becomes more challenging to be able to accruately predict how different aspects affect sales.

This report will focus on a WalMart Dataset that has Department-wise Weekly Sales of 45 WalMart stores. It will attempt to create a predictive model and also discuss the extent to which different factors affect the sales.

2. Introduction

2.1 About the Solution Environment

The authors implemented this solution in R. We have used R Markdown Report to create this document. First we explore and prepare the data set before carrying out formal statistical inferences on the dataset. We wrap the report by building a model to predict Weekly sales of the departments belonging to the 45 stores in this dataset.

2.2 About the Data

The dataset under consideration is taken from a recruitment competition WalMart ran on Kaggle between February-May 2014. Each store has multiple departments and the end requirement is to be able to predict the sales for individual departments of each store.

The training dataset has more than 400K records. The testing dataset has over 100K recrods.

2.2.1 The Challenge

The challenge is to be able to predict how different holiday price markdowns affect the various departments in the store, to model extent of impact of these markdowns.

2.3 Getting the Data

The data was download from Kaggle.

URL to the Kaggle Competition Site: https://www.kaggle.com/c/walmart-recruiting-store-sales-forecasting

The files available are the following:

2.3.1 The Data Files

Here we discuss the various CSV Files that are given by WalMart.

2.3.1.1 stores.csv

Contains size and type of 45 stores (45 records).

2.3.1.2 train.csv

Weekly sales dataset from Februray 05, 2010 to November 11, 2012. It contains the following fields:

- Store: store number
- Dept: the department number
- Date: week date
- Weekly_Sales: sales for the given department in the given store
- IsHoliday: whether the week is a special holiday week

2.3.1.3 test.csv

The dataset with similar fields as train.csv, except without Weekly_Sales. This will be used to test the model with unseen data and can be evaulated by uploading the dataset to Kaggle.

2.3.1.4 features.csv

This data file contains additional relevant information relating to the physical and business environment around the store. The fields are as follows:

- Store: store number
- Date: the week date
- Temperature: the average temperature in the region
- Fuel_Price: cost of fuel in the region
- MarkDown1-5: data related to the markdowns that Walmart is running. Markdown data is only available after November 2011 and is not available for all stores all the time. Any missing value is marked with an NA.
- CPI - the Consumer Price Index
- Unemployment - the unemployment rate
- IsHoliday - whether the week is a special holiday week

The four holidays fall inthe following weeks in the dataset:

- Super Bowl: 12-Feb-10, 11-Feb-11, 10-Feb-12, 8-Feb-13
- Labor Day: 10-Sep-10, 9-Sep-11, 7-Sep-12, 6-Sep-13
- Thanksgiving: 26-Nov-10, 25-Nov-11, 23-Nov-12, 29-Nov-13
- Christmas: 31-Dec-10, 30-Dec-11, 28-Dec-12, 27-Dec-13

2.3.2 Ingesting the Data

```r
## Ingesting the data from the Data folder
train <- read.csv("Data/train.csv")
stores <- read.csv("Data/stores.csv")
features <- read.csv("Data/features.csv")
test <- read.csv("Data/test.csv")
```

2.4 R Libraries Used

The following libraries are used in this report:

```r
# Grammar of Graphics Plotting Library
library(ggplot2)
# To use 'melt'
library(reshape2)
# to enable commas in graphs
library(scales)
# to get the month number from date variable
library(lubridate)
```

3. Stage 1: Data Exploration and Preparation

3.1 Summary Statistics

3.1.1 The Training Dataset (train)

```
str(train)
```

```
## 'data.frame':    421570 obs. of  5 variables:
##  $ Store       : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ Dept        : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ Date        : Factor w/ 143 levels "2010-02-05","2010-02-12",..: 1 2 3 4 5
6 7 8 9 10 ...
##  $ Weekly_Sales: num  24924 46039 41596 19404 21828 ...
##  $ IsHoliday   : logi  FALSE TRUE FALSE FALSE FALSE FALSE ...
```

`Date` is ingested as factor (as opposed to being ingested as date type). There are 143 dates in total.

```
## Changing the Date from "Format" type to "Date" Type
train$Date <- as.Date(train$Date)
## Getting the summary of the Data
summary(train)
```

```
##      Store           Dept            Date             Weekly_Sales
##  Min.   : 1.0   Min.   : 1.00   Min.   :2010-02-05   Min.   : -4989
##  1st Qu.:11.0   1st Qu.:18.00   1st Qu.:2010-10-08   1st Qu.:  2080
##  Median :22.0   Median :37.00   Median :2011-06-17   Median :  7612
##  Mean   :22.2   Mean   :44.26   Mean   :2011-06-18   Mean   : 15981
##  3rd Qu.:33.0   3rd Qu.:74.00   3rd Qu.:2012-02-24   3rd Qu.: 20206
##  Max.   :45.0   Max.   :99.00   Max.   :2012-10-26   Max.   :693099
##  IsHoliday
##  Mode :logical
##  FALSE:391909
##  TRUE :29661
##  NA's :0
##
##
```

There is no missing data in the dataset.

As discussed in the Introduction, this report contains data of 45 stores - represented by Store. There are a total of 99 stores in all.

The starting date for training dataset is `2010-02-05`. It starts on a `Friday`. The last date recorded in the dataset is `2012-10-26`, which is also a `Friday`. There are `994` days between them - so the data consists of a total of `143` weeks of data.

It is interesting to note that for some departments the `Weekly_Sales` are negative. Returns and special offers cause these negative sales figures.

There are no missing values in this dataset.

3.1.2 The Stores Dataset (stores)

```
## Structure of Stores Dataset
str(stores)
```

```
## 'data.frame':    45 obs. of  3 variables:
##  $ Store: int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Type : Factor w/ 3 levels "A","B","C": 1 1 2 1 2 1 2 1 2 2 ...
##  $ Size : int  151315 202307 37392 205863 34875 202505 70713 155078 125833 126
512 ...
```

```
## summary Statistics of Stores dataset
summary(stores)
```

```
##      Store     Type        Size       
##  Min.   : 1   A:22   Min.   : 34875  
##  1st Qu.:12   B:17   1st Qu.: 70713  
##  Median :23   C: 6   Median :126512  
##  Mean   :23          Mean   :130288  
##  3rd Qu.:34          3rd Qu.:202307  
##  Max.   :45          Max.   :219622  
```

No missing data.

3.1.3 The Features Dataset (features)

```
## Structure of features dataset
str(features)
```

```
## 'data.frame':    8190 obs. of  12 variables:
##  $ Store       : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ Date        : Factor w/ 182 levels "2010-02-05","2010-02-12",..: 1 2 3 4 5
6 7 8 9 10 ...
##  $ Temperature : num  42.3 38.5 39.9 46.6 46.5 ...
##  $ Fuel_Price  : num  2.57 2.55 2.51 2.56 2.62 ...
##  $ MarkDown1   : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ MarkDown2   : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ MarkDown3   : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ MarkDown4   : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ MarkDown5   : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ CPI         : num  211 211 211 211 211 ...
##  $ Unemployment: num  8.11 8.11 8.11 8.11 8.11 ...
##  $ IsHoliday   : logi  FALSE TRUE FALSE FALSE FALSE FALSE ...
```

`Date` is ingested as factor (as opposed to being ingested as date type). There are 182 dates in total. This dataset is relevant for both the `train` and the `test` dataset.

```
## Changing the Date from "Format" type to "Date" Type
features$Date <- as.Date(features$Date)
```

```
## Summary Statistics of Features Dataset
summary(features)
```

```
##      Store         Date             Temperature       Fuel_Price
##  Min.   : 1   Min.   :2010-02-05   Min.   : -7.29   Min.   :2.472
##  1st Qu.:12   1st Qu.:2010-12-17   1st Qu.: 45.90   1st Qu.:3.041
##  Median :23   Median :2011-10-31   Median : 60.71   Median :3.513
##  Mean   :23   Mean   :2011-10-31   Mean   : 59.36   Mean   :3.406
##  3rd Qu.:34   3rd Qu.:2012-09-14   3rd Qu.: 73.88   3rd Qu.:3.743
##  Max.   :45   Max.   :2013-07-26   Max.   :101.95   Max.   :4.468
##
##    MarkDown1        MarkDown2           MarkDown3           MarkDown4
##  Min.   : -2781   Min.   :  -265.76   Min.   :  -179.26   Min.   :    0.22
##  1st Qu.:  1578   1st Qu.:    68.88   1st Qu.:     6.60   1st Qu.:  304.69
##  Median :  4744   Median :   364.57   Median :    36.26   Median : 1176.42
##  Mean   :  7032   Mean   :  3384.18   Mean   :  1760.10   Mean   : 3292.94
##  3rd Qu.:  8923   3rd Qu.:  2153.35   3rd Qu.:   163.15   3rd Qu.: 3310.01
##  Max.   :103185   Max.   :104519.54   Max.   :149483.31   Max.   :67474.85
##  NA's   :4158     NA's   :5269        NA's   :4577        NA's   :4726
##    MarkDown5             CPI         Unemployment    IsHoliday
##  Min.   :  -185.2   Min.   :126.1   Min.   : 3.684   Mode :logical
##  1st Qu.:  1440.8   1st Qu.:132.4   1st Qu.: 6.634   FALSE:7605
##  Median :  2727.1   Median :182.8   Median : 7.806   TRUE :585
##  Mean   :  4132.2   Mean   :172.5   Mean   : 7.827   NA's :0
##  3rd Qu.:  4832.6   3rd Qu.:213.9   3rd Qu.: 8.567
##  Max.   :771448.1   Max.   :229.0   Max.   :14.313
##  NA's   :4140       NA's   :585     NA's   :585
```

The `features` dataset has missing variables for `Markdown1-5`, `CPI` & `Unemployment`.

3.1.4 The Test Dataset (test)

```
## Structure of test dataset
str(test)
```

```
## 'data.frame':    115064 obs. of  4 variables:
##  $ Store    : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ Dept     : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ Date     : Factor w/ 39 levels "2012-11-02","2012-11-09",..: 1 2 3 4 5 6 7
8 9 10 ...
##  $ IsHoliday: logi  FALSE FALSE FALSE TRUE FALSE FALSE ...
```

`Date` is ingested as factor (as opposed to being ingested as date type). There are 39 dates in total.

```
## Changing the Date from "Format" type to "Date" Type
test$Date <- as.Date(test$Date)
## Summary Statistics of test Dataset
summary(test)
```

```
##      Store            Dept            Date            IsHoliday
##  Min.   : 1.00   Min.   : 1.00   Min.   :2012-11-02   Mode :logical
##  1st Qu.:11.00   1st Qu.:18.00   1st Qu.:2013-01-04   FALSE:106136
##  Median :22.00   Median :37.00   Median :2013-03-15   TRUE :8928
##  Mean   :22.24   Mean   :44.34   Mean   :2013-03-14   NA's :0
##  3rd Qu.:33.00   3rd Qu.:74.00   3rd Qu.:2013-05-24
##  Max.   :45.00   Max.   :99.00   Max.   :2013-07-26
```

3.2 Data Preparation - Merging the Datasets

3.2.1 Merging Train and Stores Datasets

Since the `Type` & `Size` variables may influence the Weekly Sales, we are merging the `train` & datasets. We merge the data by `Store`.

```r
## Merging train and stores by Store
trainStoresMerge <- merge(train , stores , by = "Store")
```

3.2.2 Merging Train, Stores and Features Datasets

Since `Markdown1-5` and other variables could play an important role at predicting `Weekly_Sales`, this should be merged with the `trainStoresMerge` dataset. We merge the data by `Store` & `Date`.

```r
## Merging trainStoresMerge and features datasets
trainStoresFeaturesMerge <-
  merge( trainStoresMerge , features , by = c( "Store" , "Date" ) )
## Clearing memory - removing intermediate datasets
rm(trainStoresMerge , train)
## Fixing the name of the Column
colnames(trainStoresFeaturesMerge)[5] <- "IsHoliday"
trainStoresFeaturesMerge$IsHoliday.y <- NULL
```

3.2.3 Merging Test, Stores and Features Datasets

We similarily merge the `test`, `stores` & `features` to create the `testStoresFeaturesMerge` dataset.

```r
## Merging test and stores by Store
testStoresMerge <- merge(test , stores , by = "Store")
## Merging testStoresMerge and features datasets
testStoresFeaturesMerge <-
  merge( testStoresMerge , features , by = c( "Store" , "Date" ) )
## Clearing Memory - removing intermediate Datasets
rm( test , testStoresMerge , features )
## Fixing the name of the Column
colnames(testStoresFeaturesMerge)[5] <- "IsHoliday"
testStoresFeaturesMerge$IsHoliday.y <- NULL
```

3.3 Data Exploration

3.3.1 Total Sales Per Department in each Store

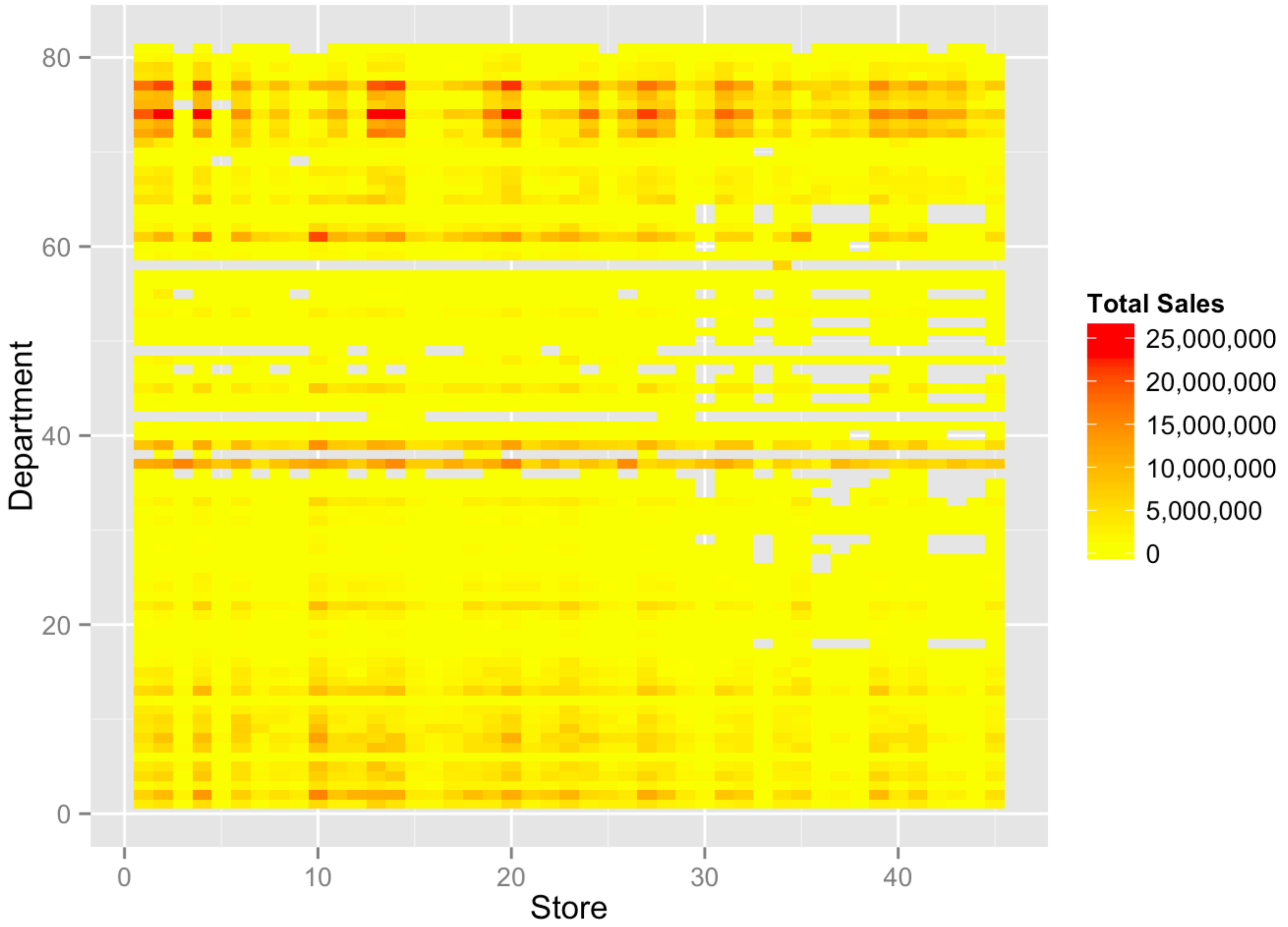
The final goal of this report is to be able to predict the weekly sales for each department in a store. First we would like to understand which departments are present in the 45 different stores and their total sales.

```
## running the sum function for each store & department
storeDeptTotalSales <- tapply(
  trainStoresFeaturesMerge$Weekly_Sales ,
  trainStoresFeaturesMerge[, c("Store","Dept")]  ,
  FUN = sum )
## Converting the matrix to a dataframe
storeDeptTotalSalesDataFrame <- as.data.frame( storeDeptTotalSales )
## Setting the Store Number into the table so we can analyze it further
storeDeptTotalSalesDataFrame$Store <-
  as.integer( rownames( storeDeptTotalSalesDataFrame ) )
## Move Store to the 1st column in the dataframe
storeDeptTotalSalesDataFrame <-
  storeDeptTotalSalesDataFrame[ , c( ncol(storeDeptTotalSalesDataFrame) , 1:nco
l(storeDeptTotalSalesDataFrame)-1 )]
## Melting the columns into rows to enable analysis
storeDeptTotalSalesDataFrame <-
  melt(storeDeptTotalSalesDataFrame , id="Store" )
## removing the NA variables - where the department does not exist in a store
storeDeptTotalSalesDataFrame <- storeDeptTotalSalesDataFrame[ complete.cases(stor
eDeptTotalSalesDataFrame),]
## Renaming the Columns in the Dataframe
colnames( storeDeptTotalSalesDataFrame )[2:3] <- c("Dept" , "TotalSales" )
## Changing the Dept Type from String to Numeric
storeDeptTotalSalesDataFrame$Dept <-
  as.integer(storeDeptTotalSalesDataFrame$Dept)
## Freeing Memory - Removing the intermediate Matrix
rm(storeDeptTotalSales)
## printing out summary statistics
summary( storeDeptTotalSalesDataFrame)
```

```
##      Store           Dept          TotalSales
##  Min.   : 1.0   Min.   : 1.00   Min.   :   -3567
##  1st Qu.:11.0   1st Qu.:19.00   1st Qu.:  137763
##  Median :22.0   Median :40.00   Median :  880317
##  Mean   :22.5   Mean   :40.49   Mean   : 2022582
##  3rd Qu.:33.0   3rd Qu.:62.00   3rd Qu.: 2609819
##  Max.   :45.0   Max.   :81.00   Max.   :26101498
```

3.3.1.1 Heatmap - Store & Department Total Sales

```
## Generating a Heatmap of the Department's Total Sales in each of 45 stores
ggplot( storeDeptTotalSalesDataFrame , aes(x = Store, y = Dept)) +
  geom_tile(aes(fill = TotalSales)) +
  scale_fill_gradient(
    low="yellow", high="red" , labels = comma , name="Total Sales") +
  scale_y_continuous(name="Department")
```



From the heatmap we can draw the following broad conclusions:

- The departments between 70-80 account for more sales than other departments
- Some departments are missing in some stores

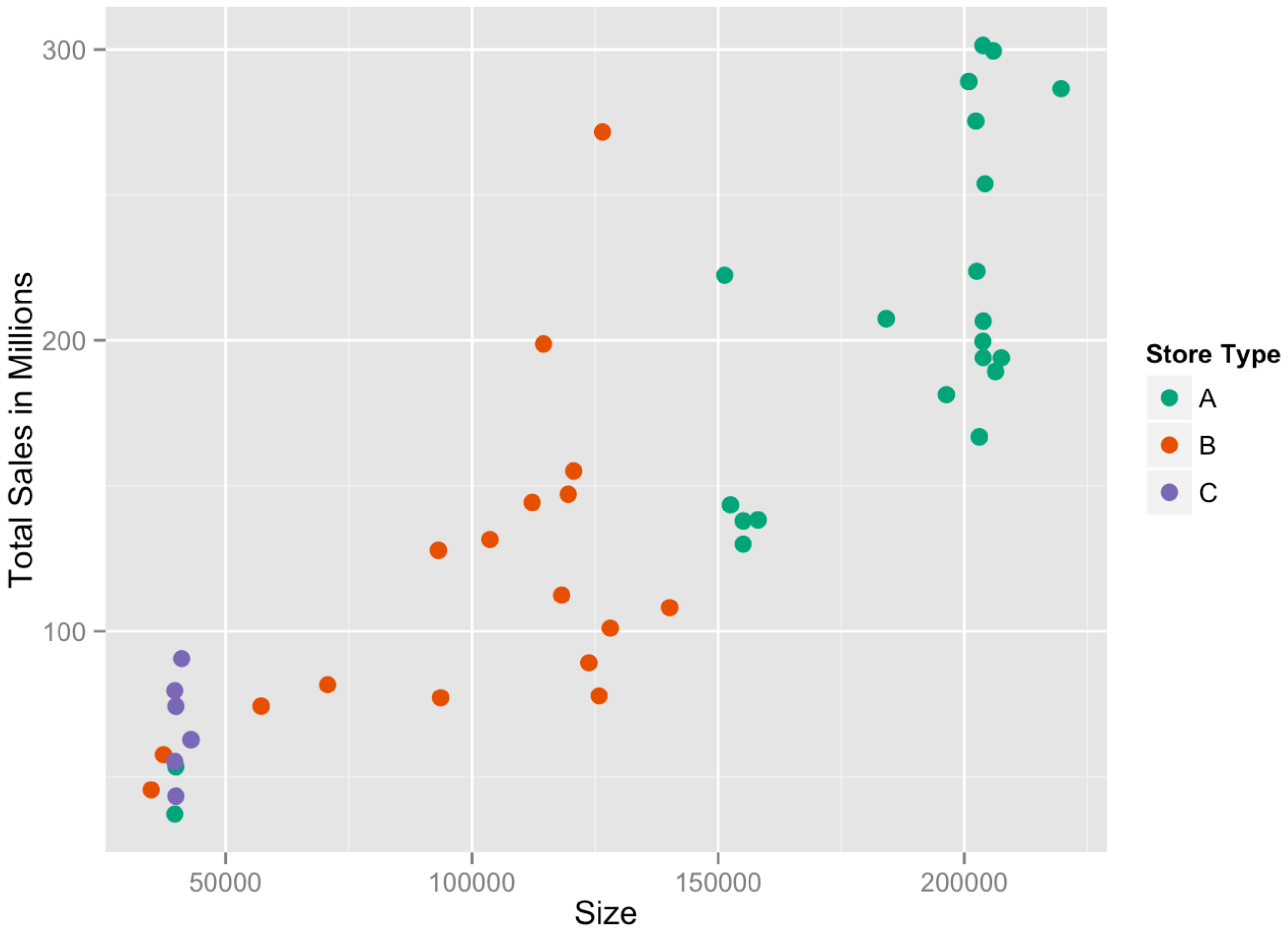
Since there are many datapoints (45 stores, each having)

3.3.2 Store Total Sales Vs. Size

Plotting the total sales of a store vs. Store Size. We first calculate the total sales per Store and plot it as a response (y-axis) to the Store size (x-axis) to understand the relationship between them.

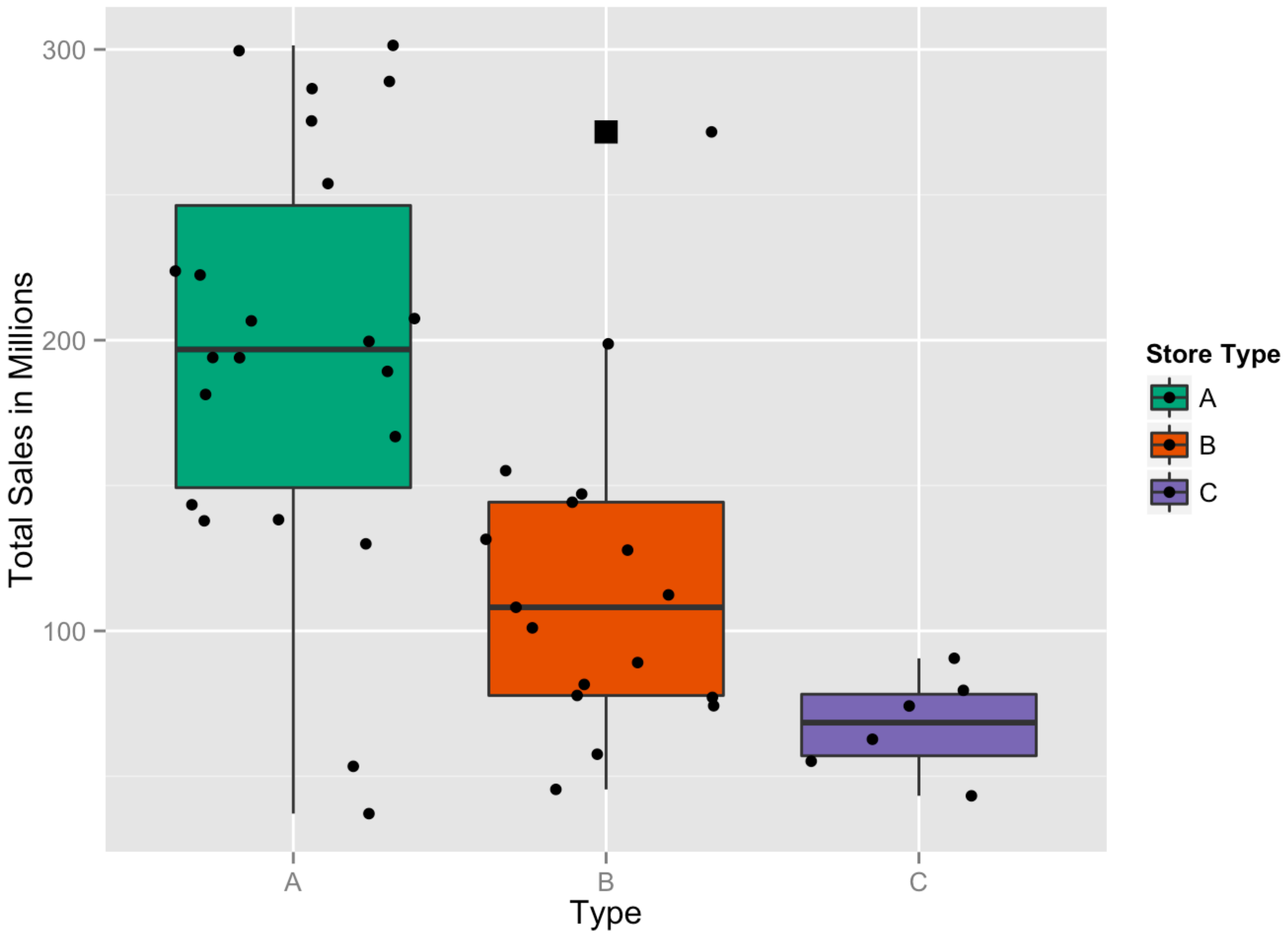
```
## Total Sales vs. Store Size - plotting the relationship
## calculating the sum of all the store sales
StoreTotalSales <-
  tapply(
    trainStoresFeaturesMerge$Weekly_Sales,
    trainStoresFeaturesMerge$Store,
    FUN = sum)
## converting the table to a DataFrame
stores$TotalSales <- StoreTotalSales
stores$TotalSalesInMillion <- stores$TotalSales/1000000
```

```
## Plotting the Total Sales vs. Store Size
ggplot( stores , aes(x=Size , y=TotalSalesInMillion , color = Type ) ) +
  geom_point( size=3) +
  scale_y_continuous(name="Total Sales in Millions" ) +
  scale_color_brewer(palette = "Dark2", name="Store Type" )
```



This plot indicates that there is a postive relationship between the size of the store and total sales. Also Type ‘A’ Stores are mostly larger stores with bigger sales and Type ‘C’ Stores are small with lower sales.

```
## box plot to show the summary statistics of the Type of Stores
ggplot(data=stores,
       aes(x=Type, y=TotalSalesInMillion, fill=Type) ) +
  geom_boxplot(outlier.shape = 15, outlier.size = 4) +
  ## to show how the individual store sales are distributed
  geom_jitter() +
  scale_y_continuous(name="Total Sales in Millions" ) +
  scale_fill_brewer(name = "Store Type" , palette = "Dark2")
```



```
## calculating the summary statistics for each Type
tabledTypeWiseSummaryStatistics <-
  tapply(stores$TotalSalesInMillion , stores$Type , summary)

## Changing the Labels of the tabled Summary Statistics for printing
attributes(tabledTypeWiseSummaryStatistics)$dimnames[[1]] <-
  c(
    "Type A Store Summary Statistics" ,
    "Type B Store Summary Statistics" ,
    "Type C Store Summary Statistics" )

## Printing Summary Statistics for each Type of Store (based on Total Sales)
tabledTypeWiseSummaryStatistics
```

```
## $`Type A Store Summary Statistics`
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   37.16  149.30  196.80  196.90  246.30  301.40
##
## $`Type B Store Summary Statistics`
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   45.48   77.79  108.10  117.70  144.30  271.60
##
## $`Type C Store Summary Statistics`
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   43.29   57.05   68.46   67.58   78.22   90.57
```

From this we can hypothesize that the `Type` of store could be an important predictor of `Weekly_Sales`.

3.3.3 Total Sales Per Week - Time Series

We discuss here the effect holidays have on Total Sales of 45 Stores.

```
## Running tapply with sum to find the total sales per week
totalSalesPerWeek <-
  tapply(
    trainStoresFeaturesMerge$Weekly_Sales ,
    trainStoresFeaturesMerge$Date ,
    FUN = sum )
## Converting table to Data Frame
totalSalesPerWeekDataFrame <- as.data.frame( totalSalesPerWeek )
## Converting date from String-Factor to Date Type
totalSalesPerWeekDataFrame$Date <-
  as.Date( rownames(totalSalesPerWeekDataFrame ) )
## Renaming the Column to "TotalSales"
colnames(totalSalesPerWeekDataFrame)[1] <- "TotalSales"
## Calculating the Total sales in Millions
totalSalesPerWeekDataFrame$TotalSalesInMillion =
  totalSalesPerWeekDataFrame$TotalSales/1000000
```

```
# function to handle lag
lagpad <- function(x, k) {
  if( k > 0 ) {
    # It should actually be NA in the rep function
    c(rep(0, k), x)[1 : length(x)]
  } else {
    # It should actually be NA in the rep function
    c(x[ (abs(k)+1) : length(x)] , rep(0, abs(k) ) )
  }
}
```

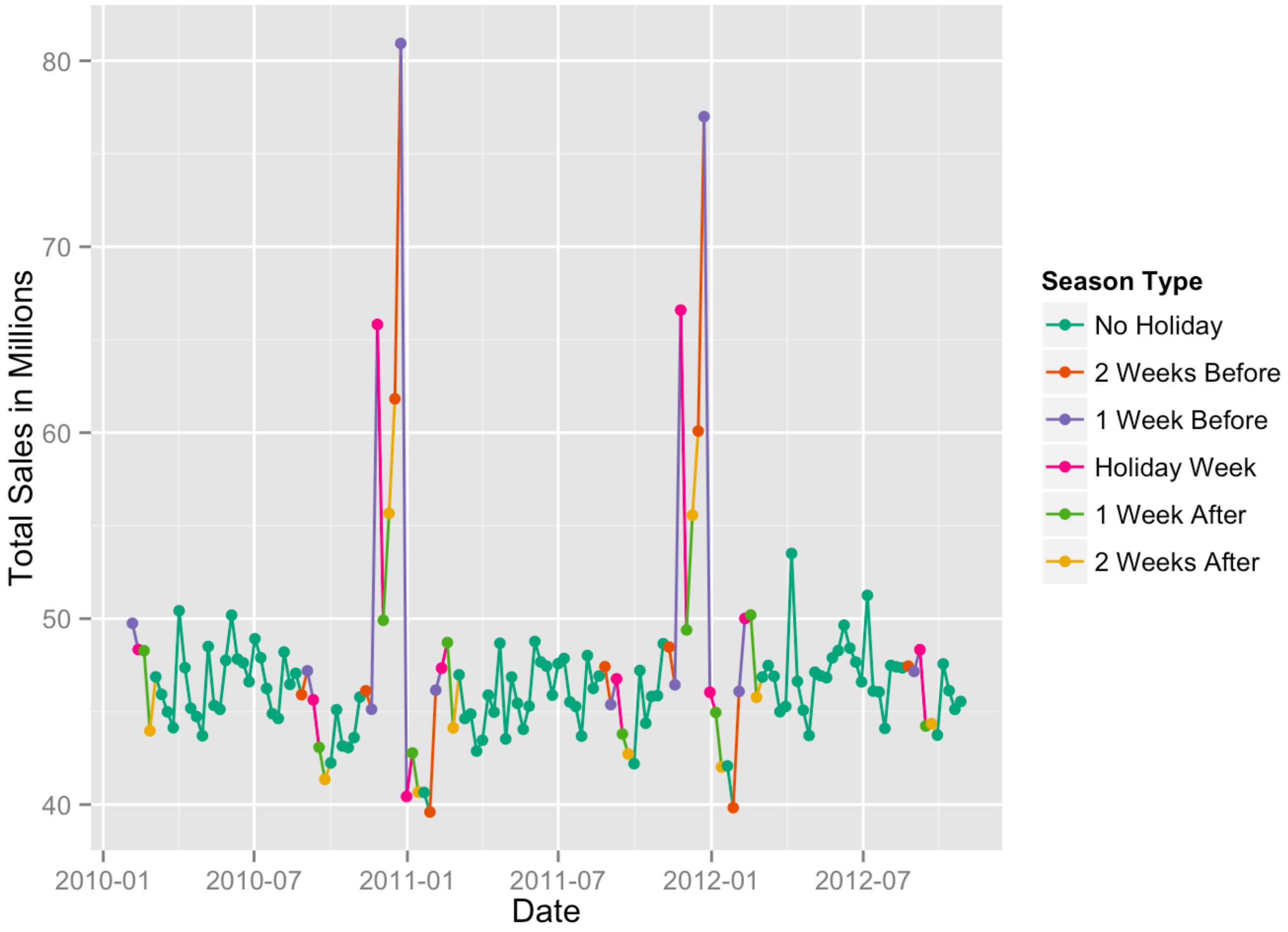
```
## Getting the holiday List
## Extracting the Holiday List
holidayDateTable <-
  table(trainStoresFeaturesMerge$Date , trainStoresFeaturesMerge$IsHoliday)
## Converting from Table to Data Frame
holidayDateTableDataFrame <- as.data.frame( holidayDateTable)
## Extracting the Holidays
holidayDateTableDataFrame <-
  subset( holidayDateTableDataFrame,holidayDateTableDataFrame$Var2==T)
## Marking the Holdiays in the dataset
holidayDateTableDataFrame$IsHoliday <-
  ifelse( holidayDateTableDataFrame$Freq >0 , 1 , 0 )
## Converting Date from String-Factor to Date Type
holidayDateTableDataFrame$Date <- as.Date( holidayDateTableDataFrame$Var1 )
# creating the holiday season - before and after the holiday week - 2 weeks
holidayDateTableDataFrame$Week1BeforeHoliday <-
  lagpad(holidayDateTableDataFrame$IsHoliday , -1)
holidayDateTableDataFrame$Week2BeforeHoliday <-
```

```
  lagpad(holidayDateTableDataFrame$IsHoliday , -2)
holidayDateTableDataFrame$Week1AfterHoliday <-
  lagpad(holidayDateTableDataFrame$IsHoliday , 1)
holidayDateTableDataFrame$Week2AfterHoliday <-
  lagpad(holidayDateTableDataFrame$IsHoliday , 2)
## Creating an ordered Holiday Season Type
holidayDateTableDataFrame$HolidaySeasonType =
  ifelse( holidayDateTableDataFrame$Week1BeforeHoliday == 1 ,
          "1 Week Before" ,
          ifelse(
            holidayDateTableDataFrame$Week2BeforeHoliday == 1 ,
            "2 Weeks Before" ,
                  ifelse(
                    holidayDateTableDataFrame$Week1AfterHoliday == 1 ,
                    "1 Week After" ,
                          ifelse(
                            holidayDateTableDataFrame$Week2AfterHoliday == 1 ,
                            "2 Weeks After" ,
                                  ifelse(
                                    holidayDateTableDataFrame$IsHoliday == 1 ,
                                          "Holiday Week" , "No Holiday" )))))
holidayDateTableDataFrame$HolidaySeasonType = factor(
  holidayDateTableDataFrame$HolidaySeasonType ,
  ordered=TRUE ,
  levels=c(
    "No Holiday" ,
    "2 Weeks Before" ,
    "1 Week Before" ,
    "Holiday Week" ,
    "1 Week After" ,
    "2 Weeks After"
    )
  )

## Creating a variable to mark if it is a Holiday season -
## that is 2 weeks before + holdiay week + 2 weeks after
holidayDateTableDataFrame$IsHolidaySeason <-
  holidayDateTableDataFrame$IsHoliday +
  holidayDateTableDataFrame$Week1BeforeHoliday * .6 +
  holidayDateTableDataFrame$Week2BeforeHoliday *.2   +
  holidayDateTableDataFrame$Week1AfterHoliday * .6 +
  holidayDateTableDataFrame$Week2AfterHoliday *.2



## Removing unneccesary Columns
holidayDateTableDataFrame$Var1 =
  holidayDateTableDataFrame$Var2 =
  holidayDateTableDataFrame$Freq = NULL
## Clearing Memory - removing intermediate Tables
rm(holidayDateTable , totalSalesPerWeek)
```

```
## Mering the sales per week and holiday list
```

```
totalSalesPerWeekDataFrame <-
  merge( totalSalesPerWeekDataFrame, holidayDateTableDataFrame , by=2)
```

```
# Plotting Sales Per Week
ggplot( totalSalesPerWeekDataFrame ,
        aes(x=Date , y=TotalSalesInMillion , color = HolidaySeasonType ) ) +
  geom_line( aes(group=1) ) +
  geom_point(size = 2) +
  scale_y_continuous(name="Total Sales in Millions" ) +
  scale_color_brewer(palette="Dark2" , name = "Season Type")
```



We can clearly see some trends here:

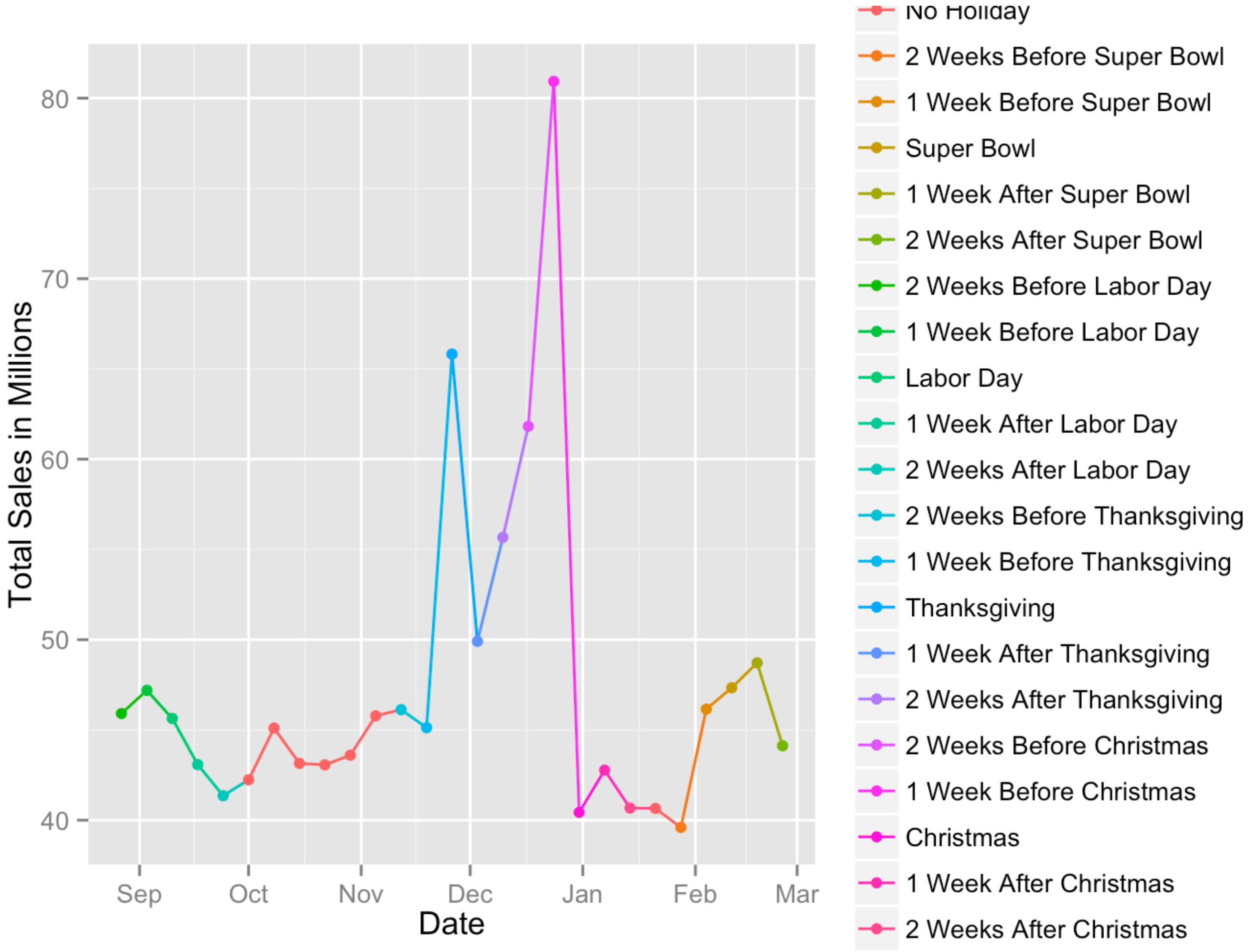
- Sales go up a week before the holiday week
- It is a repeating pattern - towards the end of the year, there is more sales - perhaps because of Thanksgiving and Christmas.
- Average Total Sales per week is between 45-50 million Dollars (for 45 stores in the dataset)
- Perhaps the Data given incorrectly marks the Holiday Week for Christmas - it is marked as the dates 27, 28, 30 & 31 across different years. But looking at the peak, the highest sales for Christmas is the week before - which is perhaps the more accurate holiday week
- To be able to make better predictors, perhaps it would help to actually create separate factors for each of the holidays
- It will be interesting to study the period between Aug 27, 2010 to Feb 25, 2011. This section has all the holidays - starting with Labor Day, Thanksgiving, Christmas & Super Bowl. Since the dataset will be smaller, we will perhaps understand the data better.

```
######################################
# Creating separate holiday dummy variables
holidayDateTableDataFrame$IsHolidayDefined <-
  ifelse(
    holidayDateTableDataFrame$IsHoliday == 1 &
      month( holidayDateTableDataFrame$Date ) == 2 ,
          2, ifelse(
            holidayDateTableDataFrame$IsHoliday == 1 &
              month( holidayDateTableDataFrame$Date ) == 9 ,
                     9 , ifelse(
                       holidayDateTableDataFrame$IsHoliday == 1 &
                         month( holidayDateTableDataFrame$Date ) == 11 ,
                                 11 , ifelse(
                                   holidayDateTableDataFrame$IsHoliday == 1 &
                                     month(
                                       holidayDateTableDataFrame$Date ) == 12 ,
                                              12 , 0 ) ) ) )
# creating the holiday season lag - before and after the holiday week - 2 weeks
holidayDateTableDataFrame$Week1BeforeHoliday <-
  lagpad(holidayDateTableDataFrame$IsHolidayDefined , -1)
holidayDateTableDataFrame$Week2BeforeHoliday <-
  lagpad(holidayDateTableDataFrame$IsHolidayDefined , -2)
holidayDateTableDataFrame$Week1AfterHoliday <-
  lagpad(holidayDateTableDataFrame$IsHolidayDefined , 1)
holidayDateTableDataFrame$Week2AfterHoliday <-
  lagpad(holidayDateTableDataFrame$IsHolidayDefined , 2)


## Creating an ordered Holiday Season Type for Super Bowl
holidayDateTableDataFrame$HolidaySeasonType <-
  ifelse(
    holidayDateTableDataFrame$Week1BeforeHoliday == 2 ,
    "1 Week Before Super Bowl" ,
          ifelse(
            holidayDateTableDataFrame$Week2BeforeHoliday == 2 ,
            "2 Weeks Before Super Bowl" ,
                  ifelse(
                    holidayDateTableDataFrame$Week1AfterHoliday == 2 ,
                    "1 Week After Super Bowl" ,
                          ifelse(
                            holidayDateTableDataFrame$Week2AfterHoliday == 2 ,
                            "2 Weeks After Super Bowl" ,
                                  ifelse(
                                    holidayDateTableDataFrame$IsHolidayDefined
                                    == 2 ,
                                    "Super Bowl" , "No Holiday" )))))

## Creating an ordered Holiday Season Type for Labor Day
holidayDateTableDataFrame$HolidaySeasonType <-
  ifelse(
    holidayDateTableDataFrame$Week1BeforeHoliday == 9 ,
    "1 Week Before Labor Day" ,
          ifelse(
            holidayDateTableDataFrame$Week2BeforeHoliday == 9 ,
            "2 Weeks Before Labor Day" ,
                  ifelse(
                    holidayDateTableDataFrame$Week1AfterHoliday == 9 ,
                    "1 Week After Labor Day" ,
                          ifelse(
                            holidayDateTableDataFrame$Week2AfterHoliday == 9 ,
                            "2 Weeks After Labor Day" ,
                                  ifelse(
                                    holidayDateTableDataFrame$IsHolidayDefined
                                    == 9 , "Labor Day" ,
                                    holidayDateTableDataFrame$HolidaySeasonType
                                    ) ) ) ) )

## Creating an ordered Holiday Season Type for Thanksgiving
holidayDateTableDataFrame$HolidaySeasonType <-
  ifelse(
    holidayDateTableDataFrame$Week1BeforeHoliday == 11 ,
    "1 Week Before Thanksgiving" ,
          ifelse(
            holidayDateTableDataFrame$Week2BeforeHoliday == 11 ,
            "2 Weeks Before Thanksgiving" ,
                  ifelse(
                    holidayDateTableDataFrame$Week1AfterHoliday == 11 ,
                    "1 Week After Thanksgiving" ,
                          ifelse(
                            holidayDateTableDataFrame$Week2AfterHoliday == 11 ,
                            "2 Weeks After Thanksgiving" ,
                                  ifelse(
                                    holidayDateTableDataFrame$IsHolidayDefined
                                    == 11 , "Thanksgiving" ,
                                    holidayDateTableDataFrame$HolidaySeasonType
                                    ) ) ) ) )

## Creating an ordered Holiday Season Type for Christmas
holidayDateTableDataFrame$HolidaySeasonType <-
  ifelse(
    holidayDateTableDataFrame$Week1BeforeHoliday == 12 ,
    "1 Week Before Christmas" ,
          ifelse(
            holidayDateTableDataFrame$Week2BeforeHoliday == 12 ,
            "2 Weeks Before Christmas" ,
                  ifelse(
                    holidayDateTableDataFrame$Week1AfterHoliday == 12 ,
                    "1 Week After Christmas" ,
                          ifelse(
                            holidayDateTableDataFrame$Week2AfterHoliday == 12 ,
                            "2 Weeks After Christmas" ,
                                  ifelse(
                                    holidayDateTableDataFrame$IsHolidayDefined
                                    == 12 , "Christmas" ,
                                    holidayDateTableDataFrame$HolidaySeasonType
                                    ) ) ) ) )

holidayDateTableDataFrame$HolidaySeasonType = factor(
  holidayDateTableDataFrame$HolidaySeasonType ,
  ordered=TRUE ,
  levels=c(
    "No Holiday" ,
    "2 Weeks Before Super Bowl" ,
    "1 Week Before Super Bowl" ,
    "Super Bowl" ,
    "1 Week After Super Bowl" ,
    "2 Weeks After Super Bowl" ,
    "2 Weeks Before Labor Day" ,
    "1 Week Before Labor Day" ,
    "Labor Day" ,
    "1 Week After Labor Day" ,
    "2 Weeks After Labor Day" ,
    "2 Weeks Before Thanksgiving" ,
    "1 Week Before Thanksgiving" ,
    "Thanksgiving" ,
    "1 Week After Thanksgiving" ,
    "2 Weeks After Thanksgiving" ,
    "2 Weeks Before Christmas" ,
    "1 Week Before Christmas" ,
    "Christmas" ,
    "1 Week After Christmas" ,
    "2 Weeks After Christmas"
  )
)
```

```
## Mering the sales per week and holiday list
totalSalesPerWeekDataFrame$HolidaySeasonType <-
  holidayDateTableDataFrame$HolidaySeasonType
```

```
## Subsetting only the holidays
totalSalesPerWeekDataFrameDuringHolidays <-
  subset( totalSalesPerWeekDataFrame ,
          totalSalesPerWeekDataFrame$Date >= '2010-08-27' &
            totalSalesPerWeekDataFrame$Date <= '2011-02-25' )
```

```
## Plotting the subset of totalSalesPerWeekDataFrame
ggplot( totalSalesPerWeekDataFrameDuringHolidays ,
        aes(x=Date , y=TotalSalesInMillion , color = HolidaySeasonType ) ) +
  geom_line( aes(group=1) ) +
  geom_point(size = 2) +
  scale_y_continuous(name="Total Sales in Millions" )
```



4. Stage 2: Formal Statistical Inferences

4.1 Do Holiday Weeks Spike Sales Up?

4.2 Do Bigger Stores contribute to Higher Sales Figures?

5. Stage 3: Linear Regression: Predicting Weekly_Sales