Final Project: Walmart Data Exploration

Stephen Kappel (spk2131), Mayank Misra (mm3557), Mandeep Singh (ms4826)

Due: December 5, 2015

Introduction

For our project, we chose to use sales data provided by Walmart as part of the *Walmart Recruiting - Store Sales Forecasting* competition on Kaggle in 2014. (See https://www.kaggle.com/c/walmart-recruiting-store-sales-forecasting.) The dataset contains weekly sales for 45 Walmart stores from 2/5/2010 to 11/1/2012. The sales are broken out by store and by department. Other attributes provide further context to the sales numbers:

- *Economic indicators:* CPI, unemployment rate, fuel prices
- *Store-specific attributes:* store size, store type, markdowns (indicator variables for five types of markdowns)
- *Other:* temperature, holiday (indicator variable)

In our analysis, sales is the outcome variable of primary interest. We aim to understand what factors impact sales. In particular, we try to answer:

1. How do economic factors – over which Walmart has no control – affect Walmart's sales? Are some departments and store types impacted more significantly than others by economic factors?
2. How well can we predict sales? In predicting sales, which predictors are most significant?
3. Do markdowns have a discernable impact on sales? Do all types of markdowns have comparable effects, or are some more effective than others?

Data Preparation

Kaggle provides the data in the form of three CSVs. To make for easier analysis, we merge the data from the CSVs into a single R data.frame. We call this combined data.frame `master`.

```
require(plyr)

get.stores <- function(){
  stores <- read.csv('../data/stores.csv')
  stores$Store <- as.factor(stores$Store)
  return(stores)
}

get.features <- function(){
  features <- read.csv('../data/features.csv')
  features$Store <- as.factor(features$Store)
  features$Date <- as.Date(features$Date)
  return(features)
}

get.train <- function(){
  train <- read.csv('../data/train.csv')
  train$Store <- as.factor(train$Store)
```

```
  train$Dept <- as.factor(train$Dept)
  train$Date <- as.Date(train$Date)
  return(train)
}

get.master <- function(){
  train <- get.train()
  stores <- get.stores()
  features <- get.features()
  master <- merge(train, stores, by='Store')
  master <- merge(master, features, by=c('Store', 'Date'))
  master$IsHoliday.y <- NULL  # IsHoliday is in features and train
  master$Sales_Millions <- master$Weekly_Sales / 10e6
  master <- rename(master, c('Type'='Store_Type', 'Size'='Store_Size',
                             'IsHoliday.x'='IsHoliday', 'Weekly_Sales'='Sales'))
  return(master)
}

master <- get.master()
```

As we do our analysis, we often find it helpful to normalize a column within the context of a store. This allows for more meaningful comparisons. Because this is a common operation, we define a function to add a nomalized column to a given data.frame for a specified column/attribute. And, we add a normalized weekly sales column.

```
add.normalized.col <- function(df, col.name){
  df$temp <- df[, col.name]
  store.ply <- ddply(df, 'Store', summarize, Mean_X=mean(temp),
                     SD_X=sd(temp))
  df <- merge(df, store.ply, by=c('Store'))
  df$Norm_X <- (df[,col.name] - df$Mean_X) / df$SD_X
  df$Mean_X <- NULL
  df$SD_X <- NULL
  df$temp <- NULL
  df <- rename(df, c('Norm_X'=paste('Norm_', col.name, sep='')))
  return(df)
}

master <- add.normalized.col(master, 'Sales')
```

It's also common for us to want to aggregate sales across some other dimension(s), so we define a function for this operation, too.

```
get.aggr.sales <- function(df, aggr.dims, filter=NULL){
  # df: master data.frame
  # aggr.dims: a list of column names to aggregate on
  # filter: NULL or c(column name to filter, value to filter to)
  if(!is.null(filter)){
    sub.df <- subset(df, df[,filter[1]] == filter[2])
  } else{
    sub.df <- df
  }
  grouped <- ddply(sub.df, aggr.dims, summarize, Sales=sum(Sales),
```

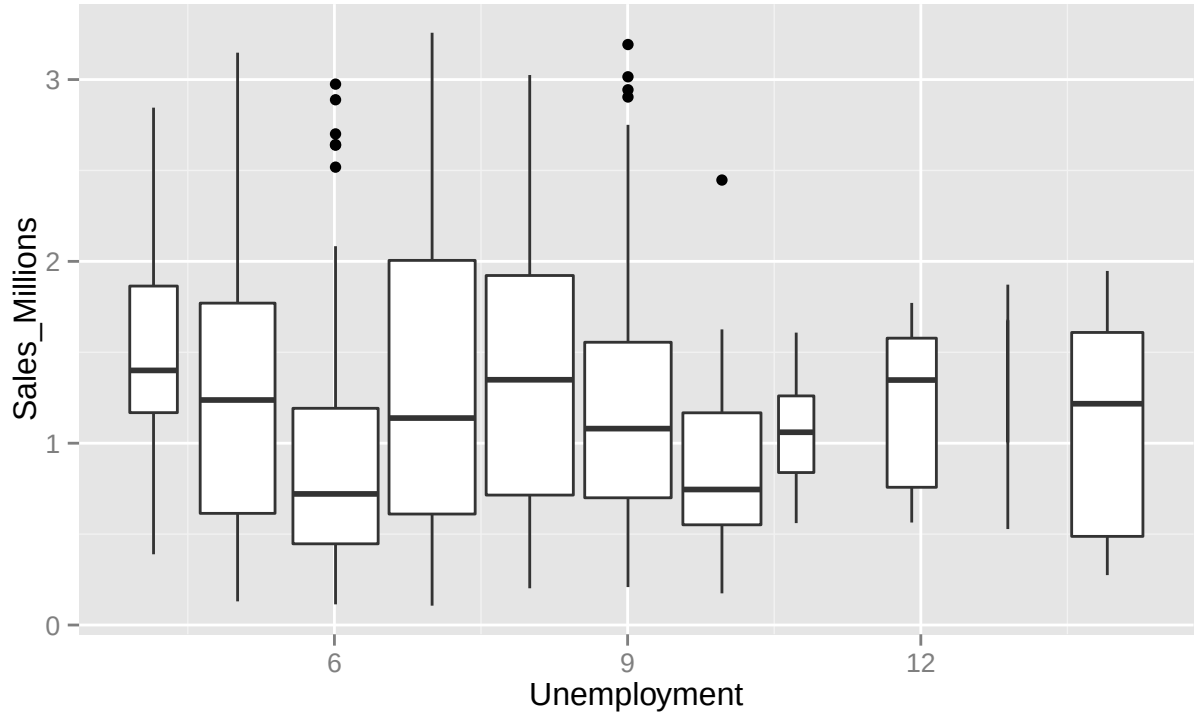
```
                    Norm_Sales=sum(Norm_Sales), Sales_Millions=sum(Sales_Millions))
  return(grouped)
}
```

Impact of economic factors

In this section we explore how economic factors (unemployment, CPI, and fuel price) relate to Walmart's sales. It's not immediately obvious how we should expect Walmart's sales to vary with economic conditions. While poor economic conditions hurt most retailers, Walmart is known for low prices. Could bad economic times drive more people to shop at Walmart?

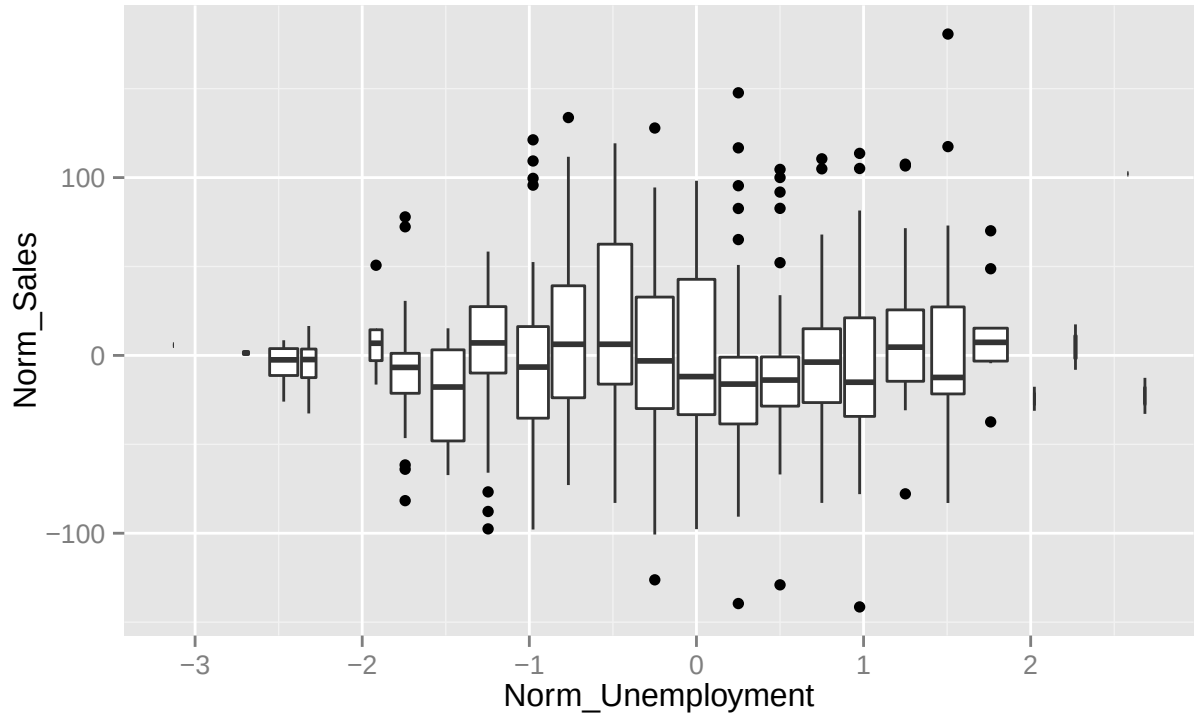
We start by creating a boxplot of sales by unemployment rate.

```
require(ggplot2)
sales.by.unemployment <- get.aggr.sales(master, c('Store', 'Unemployment'))
ggplot(data=sales.by.unemployment, aes(x=Unemployment, y=Sales_Millions,
  group=round_any(Unemployment, 1.0))) + geom_boxplot()
```



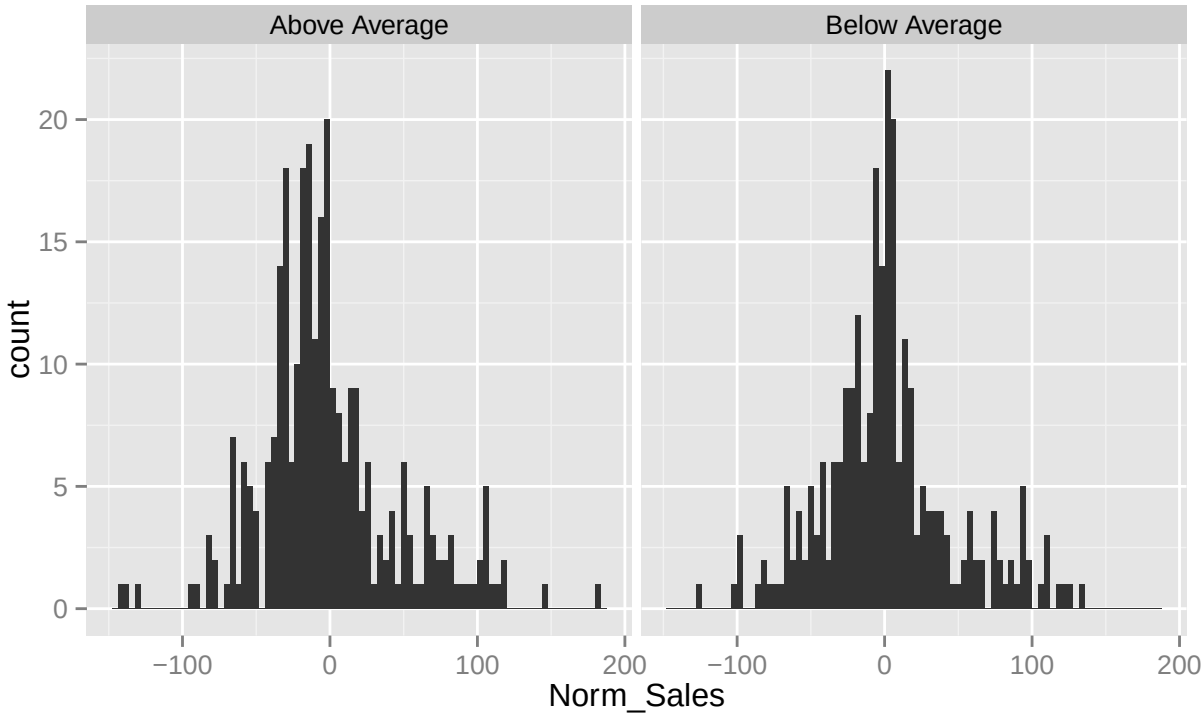
This plot don't show much of a relationship, but we haven't done any normalization, so this probably isn't a meaningful representation of the relationship. We normalize sales and unemployment rate by store and create an update boxplot.

```
master <- add.normalized.col(master, 'Unemployment')
sales.by.norm.unemployment <- get.aggr.sales(master, c('Store', 'Norm_Unemployment'))
ggplot(data=sales.by.norm.unemployment, aes(x=Norm_Unemployment, y=Norm_Sales,
  group=round_any(Norm_Unemployment, 0.25))) + geom_boxplot()
```



In this most recent plot, there appears to be below-average sales when there is above-average unemployment. Let's create a pair of histograms showing number of weeks by normalized sales - one histogram for weeks with below average unemployment and on histogram for weeks with above average unemployment. We want to see if these distributions are noticably different. We also want to see the shape of these distributions to see if the data is normally distributed and a t-test would be reasonable.

```
sales.by.norm.unemployment$Unemployment_Category <- ifelse(
  sales.by.norm.unemployment$Norm_Unemployment < 0, 'Below Average', 'Above Average')
ggplot(data=sales.by.norm.unemployment, aes(x=Norm_Sales)) + geom_histogram(binwidth=4) +
  facet_wrap(~Unemployment_Category)
```



The relationship between unemployment and sales isn't too clear from this plot, but the populations doappear to be roughly normal. We now run an F-test to check if the populations have equal variances (as assumed by a two-sample t-test).

```
above <- subset(sales.by.norm.unemployment, Unemployment_Category == 'Above Average')
below <- subset(sales.by.norm.unemployment, Unemployment_Category == 'Below Average')
var.test(above$Norm_Sales, below$Norm_Sales)
```

```
##
##  F test to compare two variances
##
## data:  above$Norm_Sales and below$Norm_Sales
## F = 1.0983, num df = 283, denom df = 255, p-value = 0.4447
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  0.8634435 1.3949339
## sample estimates:
## ratio of variances
##           1.098305
```

We don't reject the null hypothesis that the populations have equal variance, so we proceed with the t-test assuming equal variances. The null hypothesis of the t-test is that the mean normalized sales during weeks with above average unemployment rates is equal to the mean normalized sales during weeks with below average umemployment rates.

```
t.test(above$Norm_Sales, below$Norm_Sales, var.equal=TRUE)
```

```
##
```

```
##  Two Sample t-test
##
## data:  above$Norm_Sales and below$Norm_Sales
## t = -1.0357, df = 538, p-value = 0.3008
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -11.857944   3.670536
## sample estimates:
## mean of x mean of y
## -1.940719  2.152985
```

We can only reject the null hypothesis with significance of 0.30. If we hadn't been able to assume equal variance, and we performed a Welsh two-sample t-test, would the results have been much different?

```
t.test(above$Norm_Sales, below$Norm_Sales, var.equal=FALSE)
```

```
##
##  Welch Two Sample t-test
##
## data:  above$Norm_Sales and below$Norm_Sales
## t = -1.0382, df = 536.249, p-value = 0.2996
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -11.83913   3.65172
## sample estimates:
## mean of x mean of y
## -1.940719  2.152985
```

The results are nearly identical. Now, we drop our assumption of normality, and perform a non-parametric test for the equality of the medians. That is, our null hypothesis is that the median normalized sales during weeks with above average unemployment rates is equal to the median normalized sales during weeks with below average umemployment rates.

```
wilcox.test(Norm_Sales ~ Unemployment_Category, data=sales.by.norm.unemployment)
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  Norm_Sales by Unemployment_Category
## W = 32905, p-value = 0.05695
## alternative hypothesis: true location shift is not equal to 0
```

TODO: We repeat the same procedure for CPI and fuel price. The results are summarized in the table below:

TODO: Are some departments more elastic than others? We might expect the electronics department to be more heavily impacted than the food and clothing departments, for instance. *Perhaps we can use an ANOVA here?

Sales predictions

formula <- Norm_Weekly_Sales ~ Norm_Unemployment + IsHoliday*Store_Type + Store_Size TODO: incorporate dates into model - month indicators, lag variables, autoregressive varaibles, trend variables

TODO: linear regression model

TODO: stepwise selection to find most significant predictors

TODO: tree-based models

TODO: fit a spline or LOESS curve

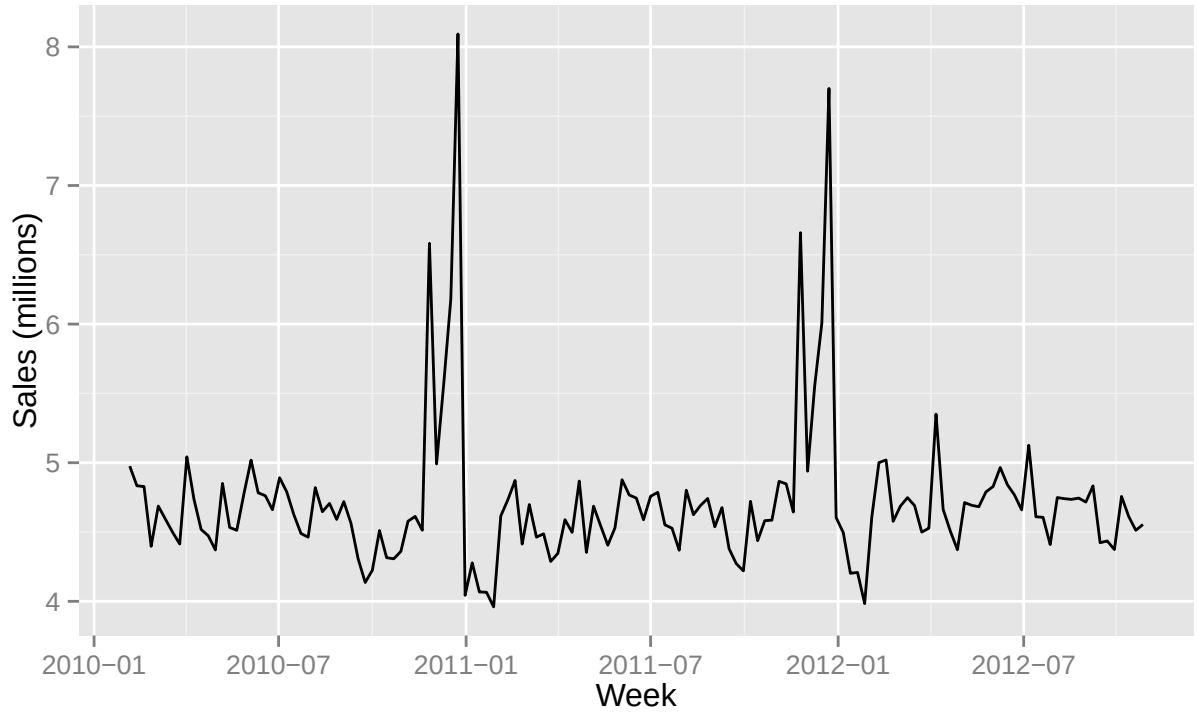
Impact of markdowns

TODO

Appendix

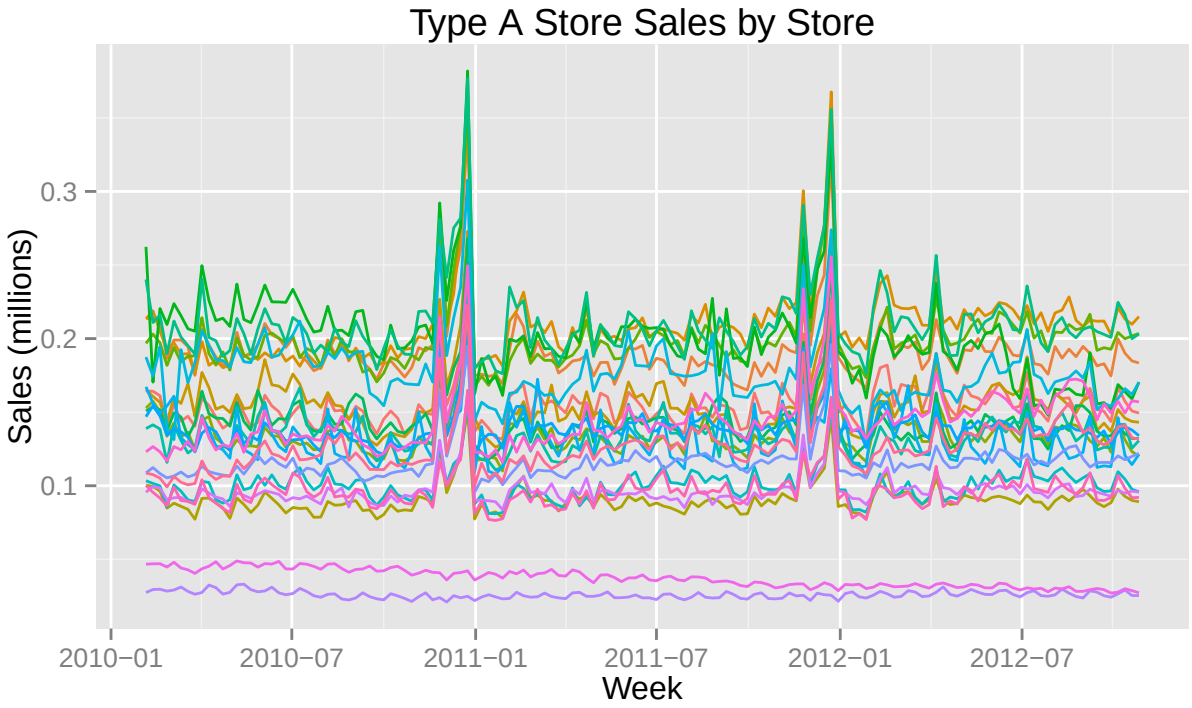
As we explored the dataset, we created many plots and computed many intermediate statistics that did not fit directly into our core findings. We share some of this work here.

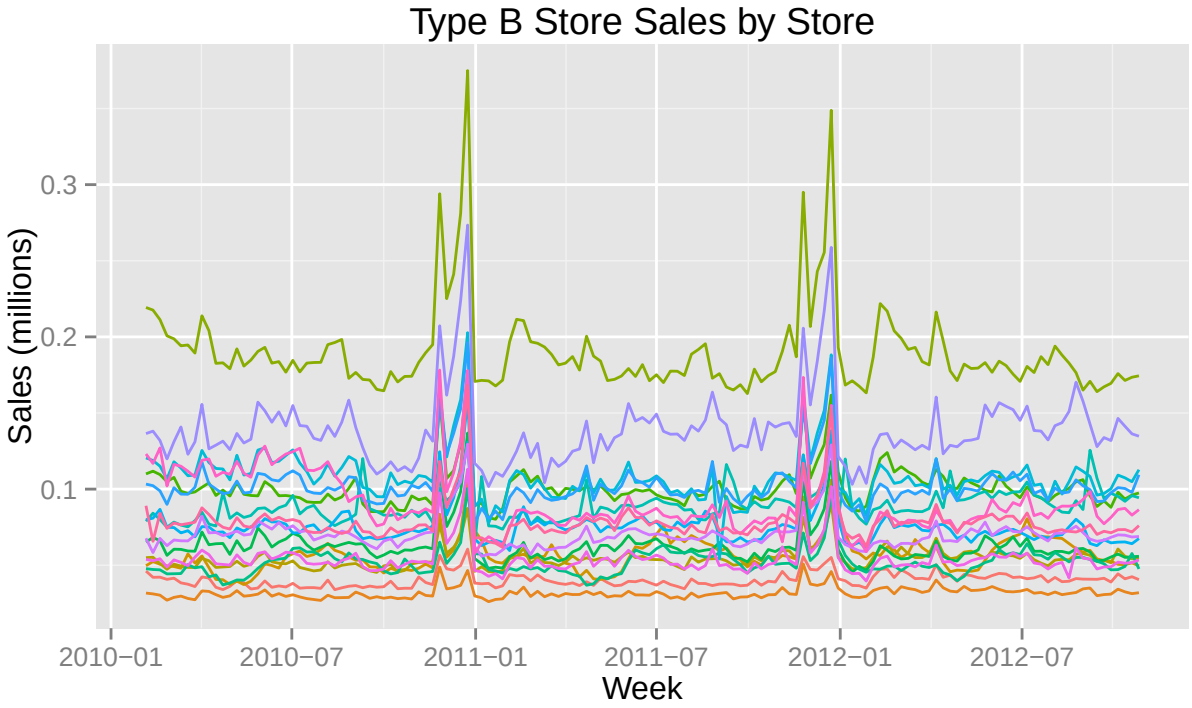
```
# plot sales by week
sales.by.week <- get.aggr.sales(master, c('Date'))
labels <- labs(x='Week', y='Sales (millions)')
ggplot(data=sales.by.week, aes(x=Date, y=Sales_Millions)) + geom_line() + labels
```

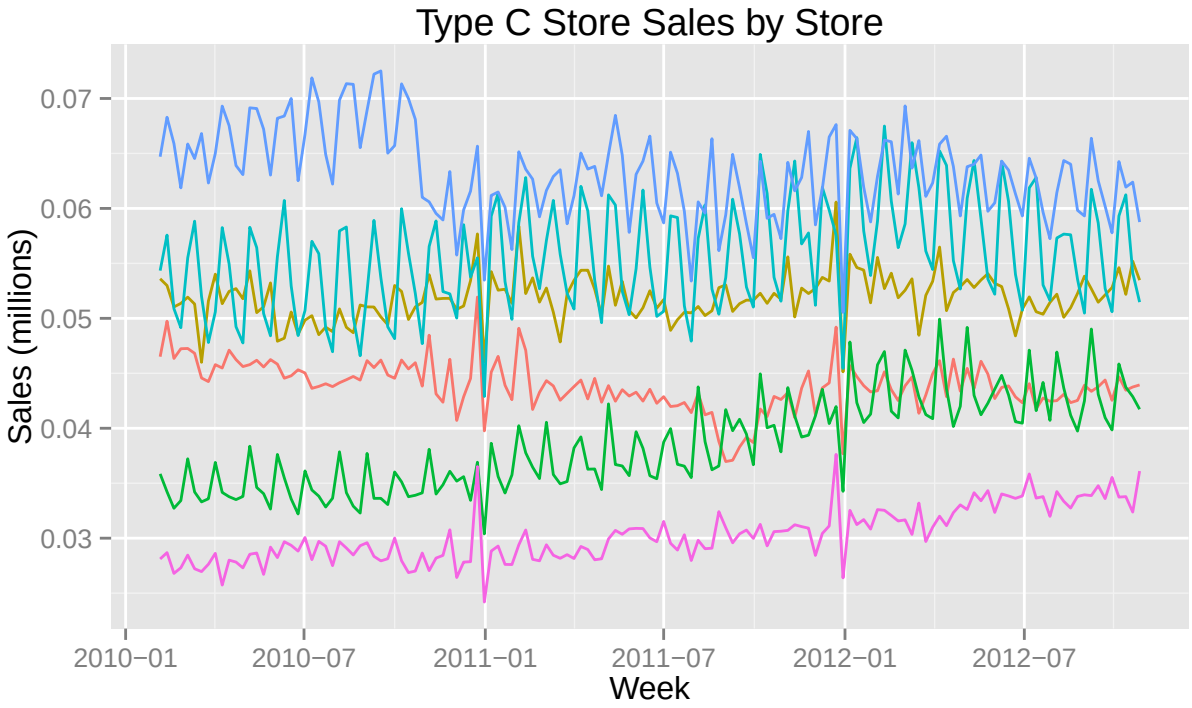



```
# plot sales by week by store type
sales.by.type <- get.aggr.sales(master, c('Store_Type','Date'))
ggplot(data=sales.by.type, aes(x=Date, y=Sales_Millions, group=Store_Type, color=Store_Type)) + geom_li
```


```
# by store (by type)
for (store.type in c('A', 'B', 'C')){
  sales.data <- get.aggr.sales(master, c('Store', 'Date'), c('Store_Type', store.type))
  print(ggplot(data=sales.data, aes(x=Date, y=Sales_Millions, group=Store, color=Store)) + geom_line() +
}
```

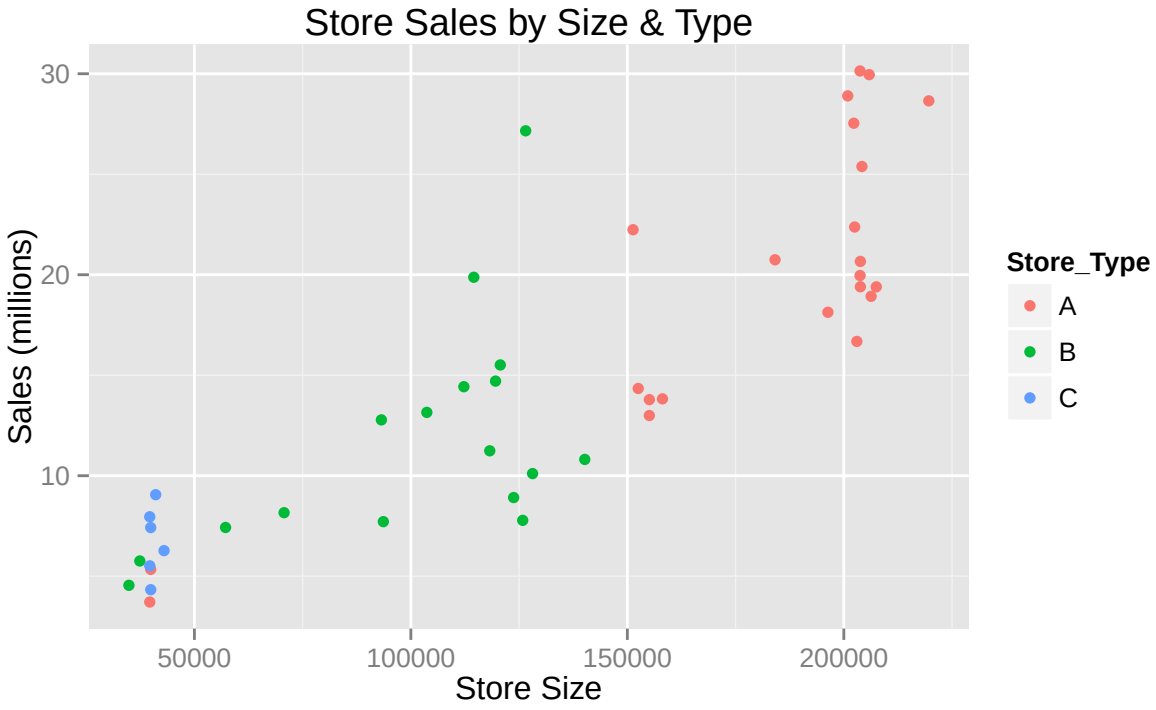

Type A Store Sales by Store
Sales (millions)
0.3
0.2
0.1
2010−01
2010−07
2011−01
2011−07
2012−01
2012−07
Week


Type B Store Sales by Store
Sales (millions)
0.3
0.2
0.1
2010−01
2010−07
2011−01
2011−07
2012−01
2012−07
Week



We can see that Type C stores are less impacted by the Holiday season than the other types of stores. We can also clearly see that most stores follow a similar seasonal patterns. Also, we see that some stores have Sales trending upward or downward over the span.

```
sales.by.size <- get.aggr.sales(master, c('Store', 'Store_Type', 'Store_Size'))
ggplot(data=sales.by.size, aes(x=Store_Size, y=Sales_Millions, color=Store_Type)) +
  geom_point() +
  labs(x='Store Size', y='Sales (millions)', title='Store Sales by Size & Type')
```



We see a clear relationship between increasing store size and increasing average store sales in this plot. We also see that the three store types are also closely associated with store sizes. Store type A generally equates to larger stores with higher sales volumes. Store type A has the smaller stores with smaller sales volumes. Store type B fits somewhere between A and C.