ADS 506 Final Project Code

Abanather Negusu & Claire Phibbs

2022-11-19

Appendix Code

Loading in Libraries

```
# cleaning the memory
rm(list = ls())
# libraries
library(ggplot2)

library(tidyr)
library(plyr)
library(dplyr)

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:plyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(forecast)

## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo

library(lubridate)

##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(data.table)

##
## Attaching package: 'data.table'

## The following objects are masked from 'package:lubridate':
##
##     hour, isoweek, mday, minute, month, quarter, second, wday, week,
##     yday, year

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

#install.packages('corrplot')
library(corrplot)

## corrplot 0.91 loaded

library(chron)

##
## Attaching package: 'chron'

## The following objects are masked from 'package:lubridate':
##
##     days, hours, minutes, seconds, years

library(fpp2)

## ── Attaching packages ─────────────────────────────────────────────── fpp2
2.4 ──

## ✓ fma       2.4     ✓ expsmooth 2.3

##

require(gridExtra)

## Loading required package: gridExtra

##
## Attaching package: 'gridExtra'

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

Loading in all 3 data sets

```
Sales_df <- read.csv("/Users/clairephibbs/Desktop/ADS 506 Applied Time Series
Analysis/Final Project/sales data-set.csv")

Feature_df <- read.csv("/Users/clairephibbs/Desktop/ADS 506 Applied Time Seri
```

```
es Analysis/Final Project/Features data set.csv")

Stores_df <- read.csv("/Users/clairephibbs/Desktop/ADS 506 Applied Time Serie
s Analysis/Final Project/stores data-set.csv")
```

Sales Data

```
# change the variable into appropriate type
Sales_df <-
  Sales_df %>%
  mutate(Store = as.factor(Store),
         Dept = as.factor(Dept),
         IsHoliday = as.factor(IsHoliday),
         Date = as.Date(Date, "%d/%m/%Y"))

# removing the nenagive sales
Sales_df <- Sales_df %>%
  filter(Weekly_Sales >= 0)
```

Feature Data

```
Feature_df <-
  Feature_df %>%
  mutate(Store = as.factor(Store),
         IsHoliday = as.factor(IsHoliday),
         Date = as.Date(Date, "%d/%m/%Y"))
```

Combining Sales and Feature Data

```
df <- Sales_df %>%inner_join(Feature_df) %>%
  select(c(1:7, 13:14), Weekly_Sales)

## Joining, by = c("Store", "Date", "IsHoliday")

# attaching the data

attach(df)

# reformatting date to 2 digit years
df$Date <- as.character(df$Date, format = '%m/%d/%y')
```

Store Data

```
Stores_df <-
  Stores_df %>%
  mutate(Store = as.factor(Store))
```

Exploratory Data Analysis (EDA):

From this point forwards only working with Store 1 data.

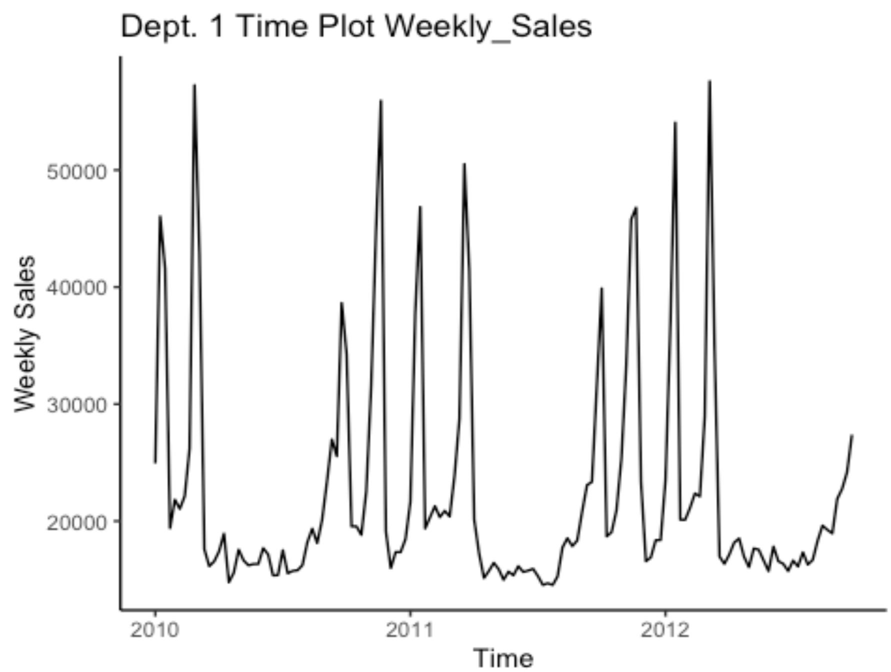
Time Series Objects for the Weekly_Sales

```
# separating departments of store 1 and creating time series objects
store_1.1 <- df[1:143, ]
```

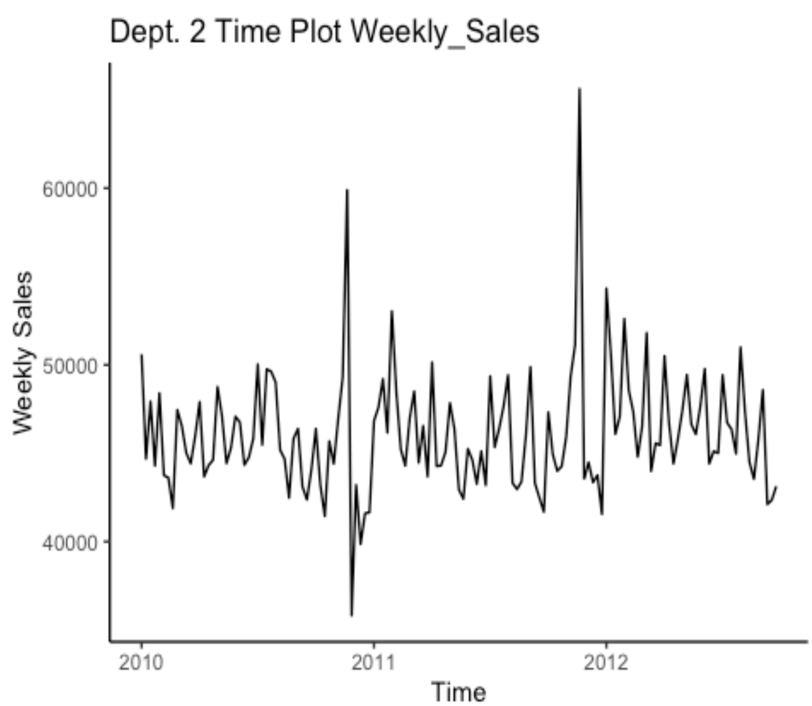
```
store_1.1_ts <- ts(store_1.1$Weekly_Sales, start = c(2010), frequency = 52)

store_1.2 <- df[144:286, ]
store_1.2_ts <- ts(store_1.2$Weekly_Sales, start = c(2010), frequency = 52)

store_1.3 <- df[287:429, ]
store_1.3_ts <- ts(store_1.3$Weekly_Sales, start = c(2010), frequency = 52)

store_1.4 <- df[430:572, ]
store_1.4_ts <- ts(store_1.4$Weekly_Sales, start = c(2010), frequency = 52)

store_1.5 <- df[573:715, ]
store_1.5_ts <- ts(store_1.5$Weekly_Sales, start = c(2010), frequency = 52)

store_1.6 <- df[715:857, ]
store_1.6_ts <- ts(store_1.6$Weekly_Sales, start = c(2010), frequency = 52)

store_1.7 <- df[858:1000, ]
store_1.7_ts <- ts(store_1.7$Weekly_Sales, start = c(2010), frequency = 52)

store_1.8 <- df[1001:1143, ]
store_1.8_ts <- ts(store_1.8$Weekly_Sales, start = c(2010), frequency = 52)

store_1.9 <- df[1144:1286, ]
store_1.9_ts <- ts(store_1.9$Weekly_Sales, start = c(2010), frequency = 52)

store_1.10 <- df[1287:1429, ]
store_1.10_ts <- ts(store_1.10$Weekly_Sales, start = c(2010), frequency = 52)

store_1.11 <- df[1430:1572, ]
store_1.11_ts <- ts(store_1.11$Weekly_Sales, start = c(2010), frequency = 52)

store_1.12 <- df[1573:1715, ]
store_1.12_ts <- ts(store_1.12$Weekly_Sales, start = c(2010), frequency = 52)

store_1.13 <- df[1716:1858, ]
store_1.13_ts <- ts(store_1.13$Weekly_Sales, start = c(2010), frequency = 52)
```

Time Plots of Store 1 Department 1-13 Weekly Sales

```
autoplot(store_1.1_ts) +
  labs(title = "Dept. 1 Time Plot Weekly_Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```

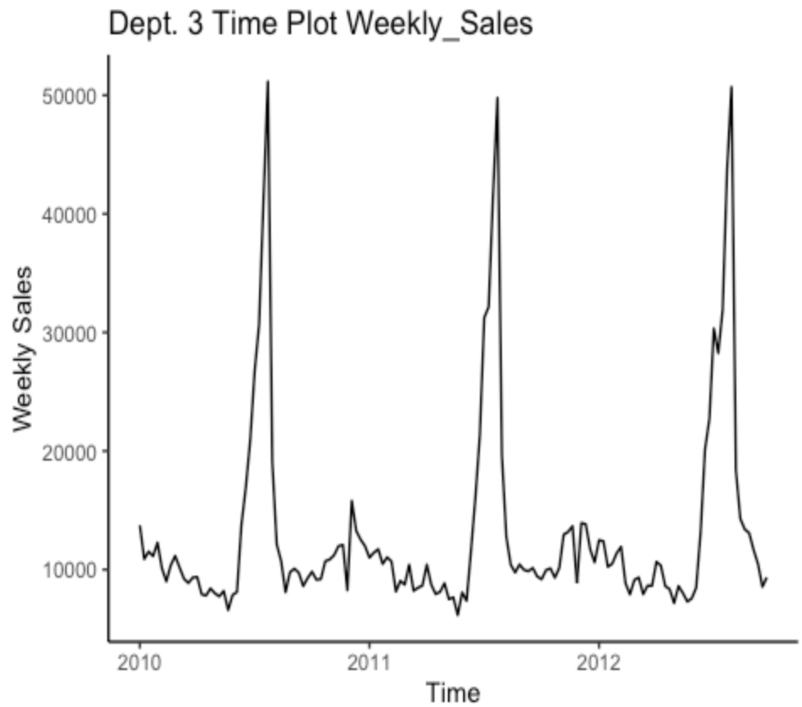



```
autoplot(store_1.2_ts) +
  labs(title = "Dept. 2 Time Plot Weekly_Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```

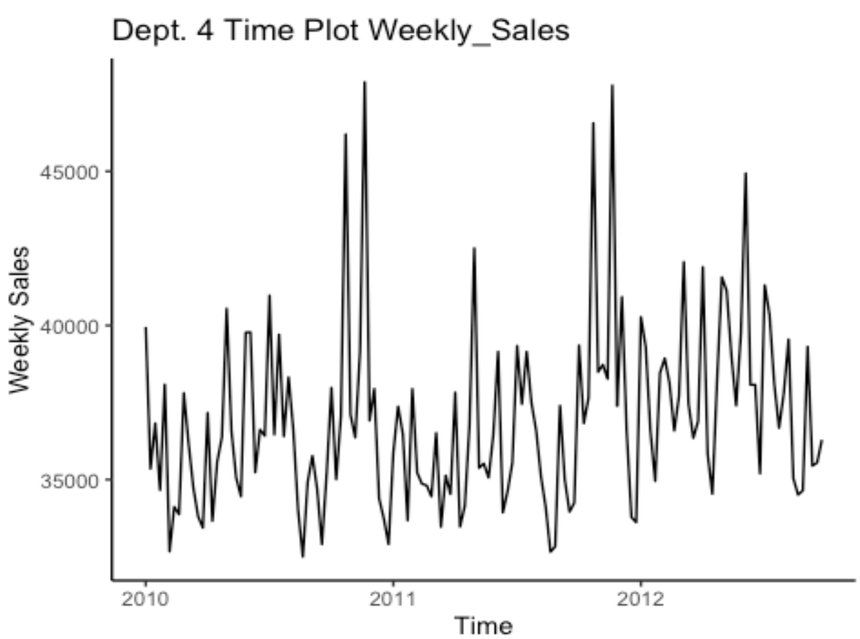



```
autoplot(store_1.3_ts) +
  labs(title = "Dept. 3 Time Plot Weekly_Sales",
```

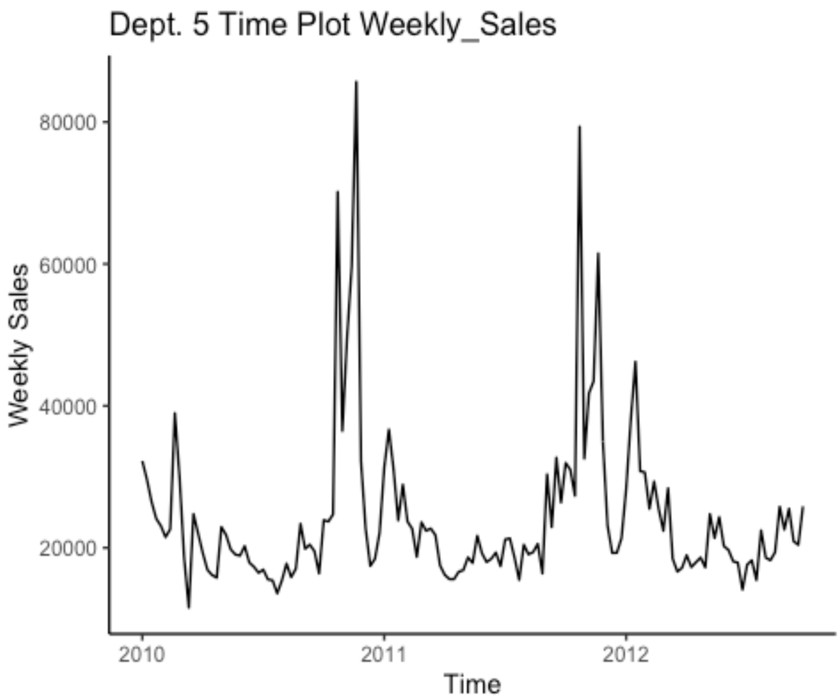
```
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```



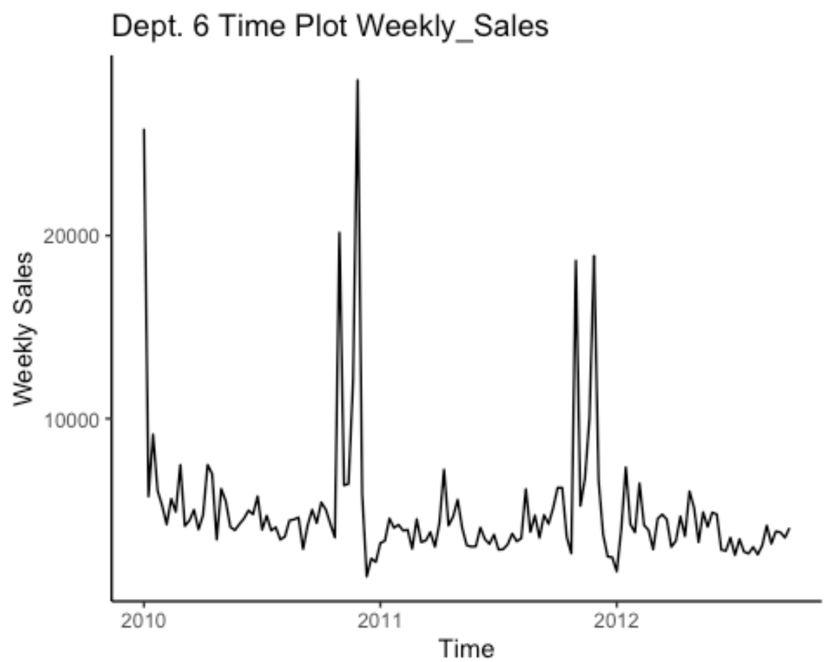

```
autoplot(store_1.4_ts) +
  labs(title = "Dept. 4 Time Plot Weekly_Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```



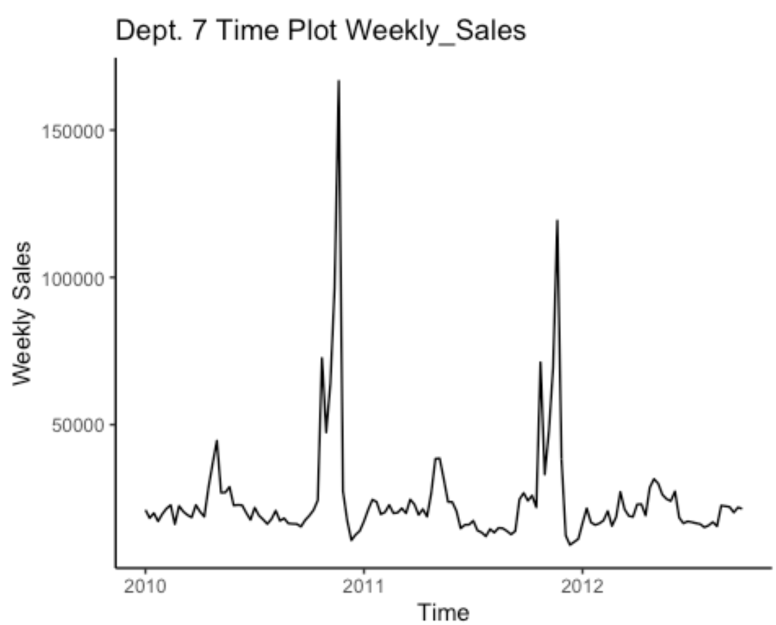
```
autoplot(store_1.5_ts) +
  labs(title = "Dept. 5 Time Plot Weekly_Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```



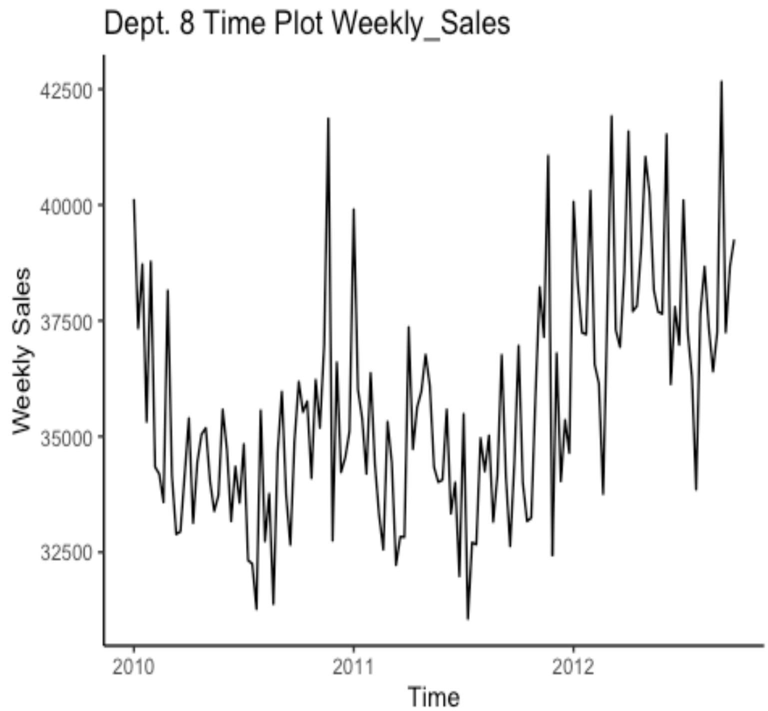

```
autoplot(store_1.6_ts) +
  labs(title = "Dept. 6 Time Plot Weekly_Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```



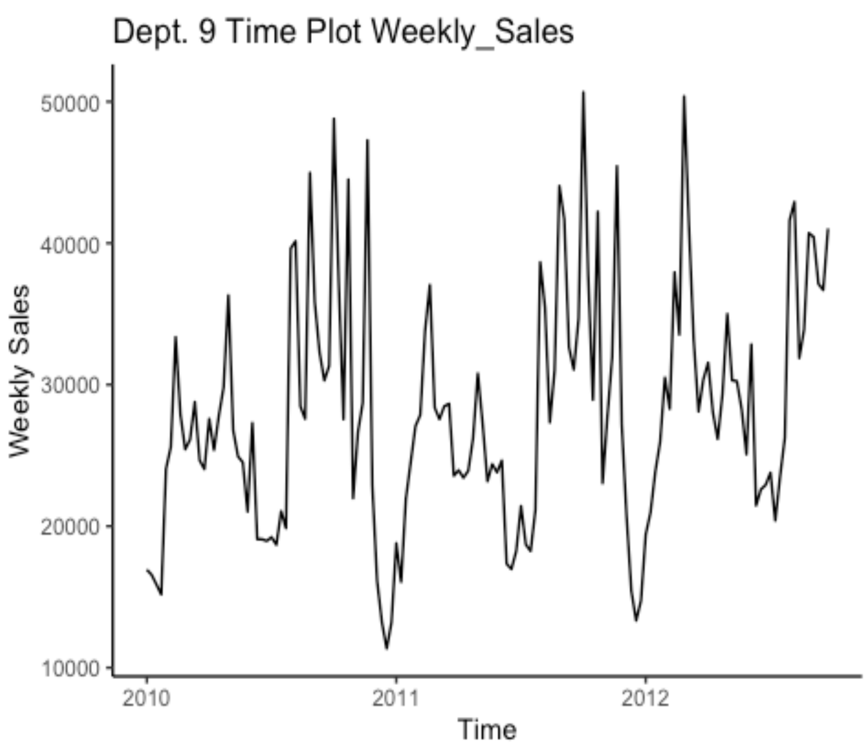

```
autoplot(store_1.7_ts) +
  labs(title = "Dept. 7 Time Plot Weekly_Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```



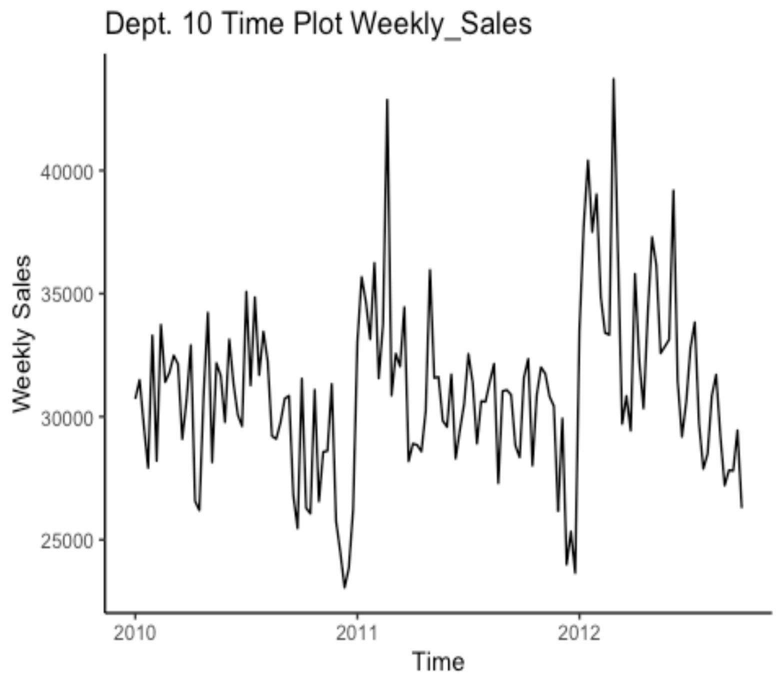

```
autoplot(store_1.8_ts) +
  labs(title = "Dept. 8 Time Plot Weekly_Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```



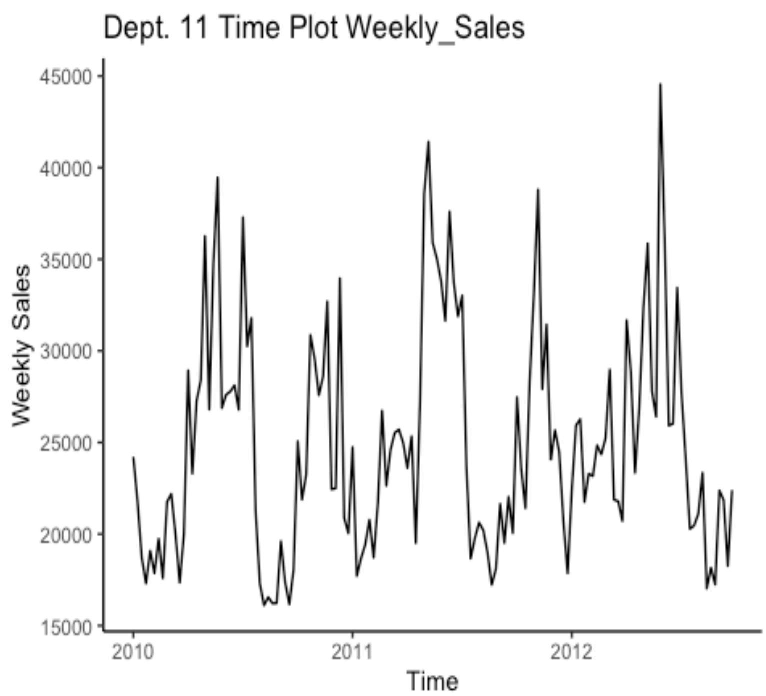

```r
autoplot(store_1.9_ts) +
  labs(title = "Dept. 9 Time Plot Weekly_Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```



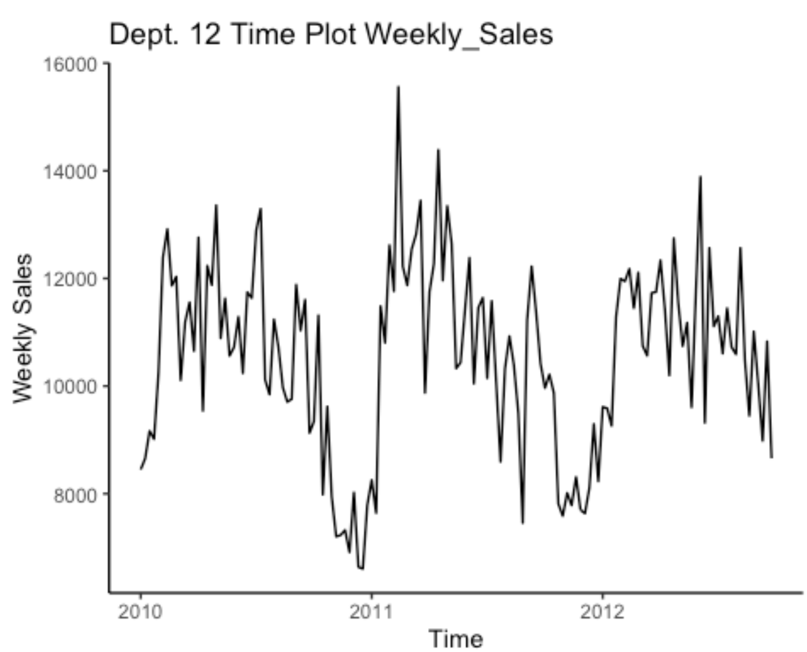
```r
autoplot(store_1.10_ts) +
  labs(title = "Dept. 10 Time Plot Weekly_Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```




```r
autoplot(store_1.11_ts) +
  labs(title = "Dept. 11 Time Plot Weekly_Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```

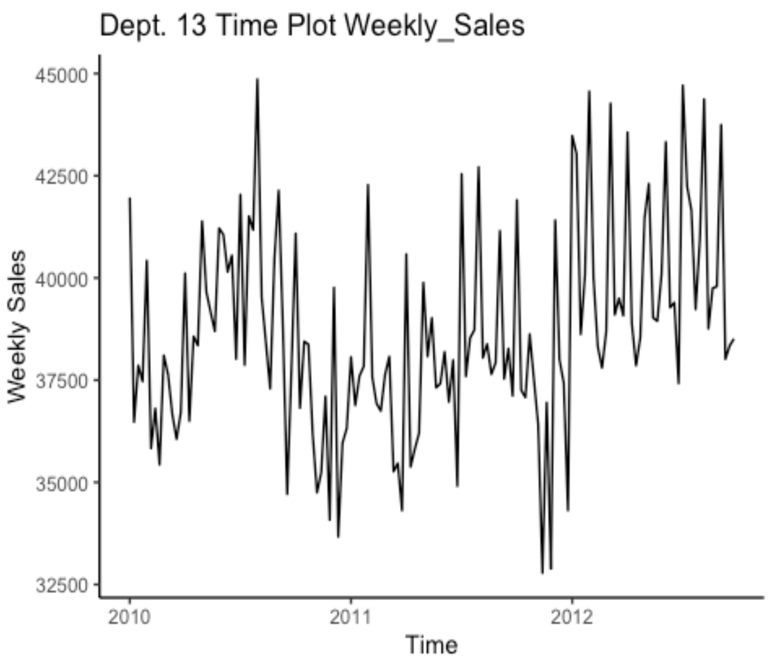



```
autoplot(store_1.12_ts) +
  labs(title = "Dept. 12 Time Plot Weekly_Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```



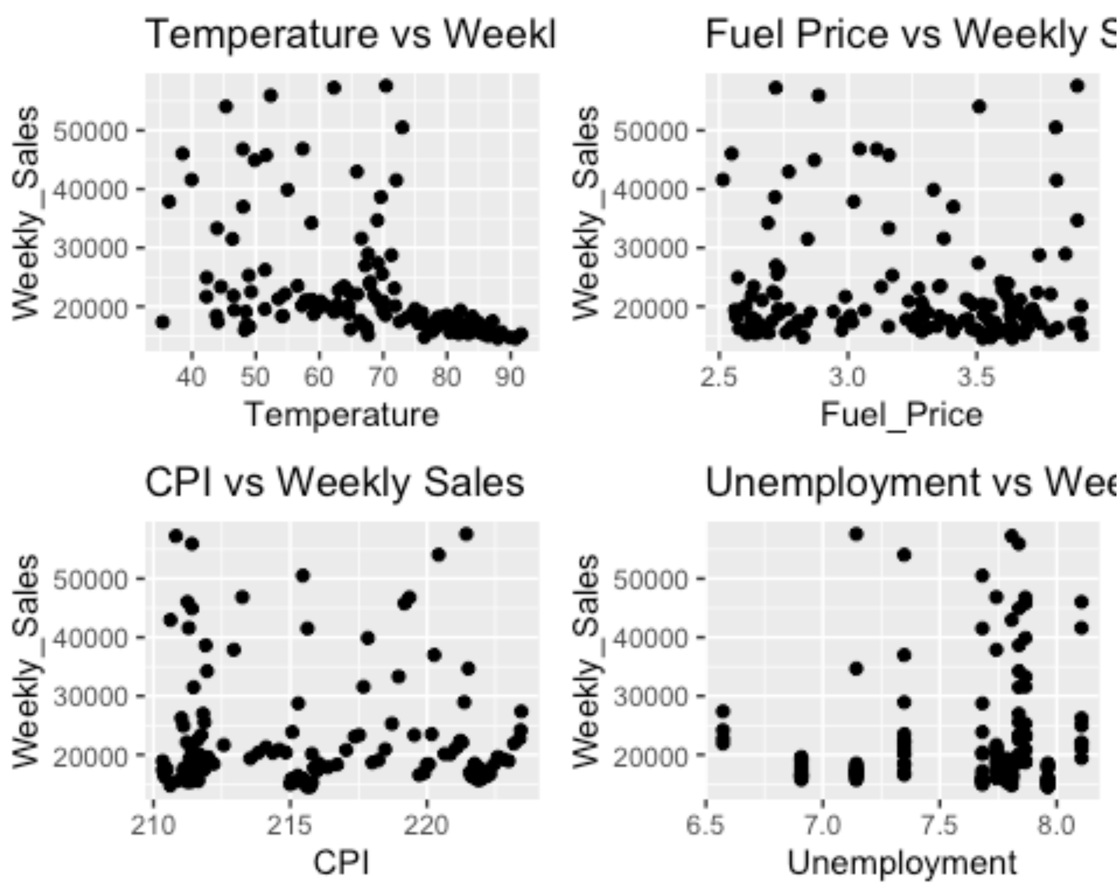

```
autoplot(store_1.13_ts) +
  labs(title = "Dept. 13 Time Plot Weekly_Sales",
       x = "Time",
```

```
      y = "Weekly Sales") +
    theme_classic()
```

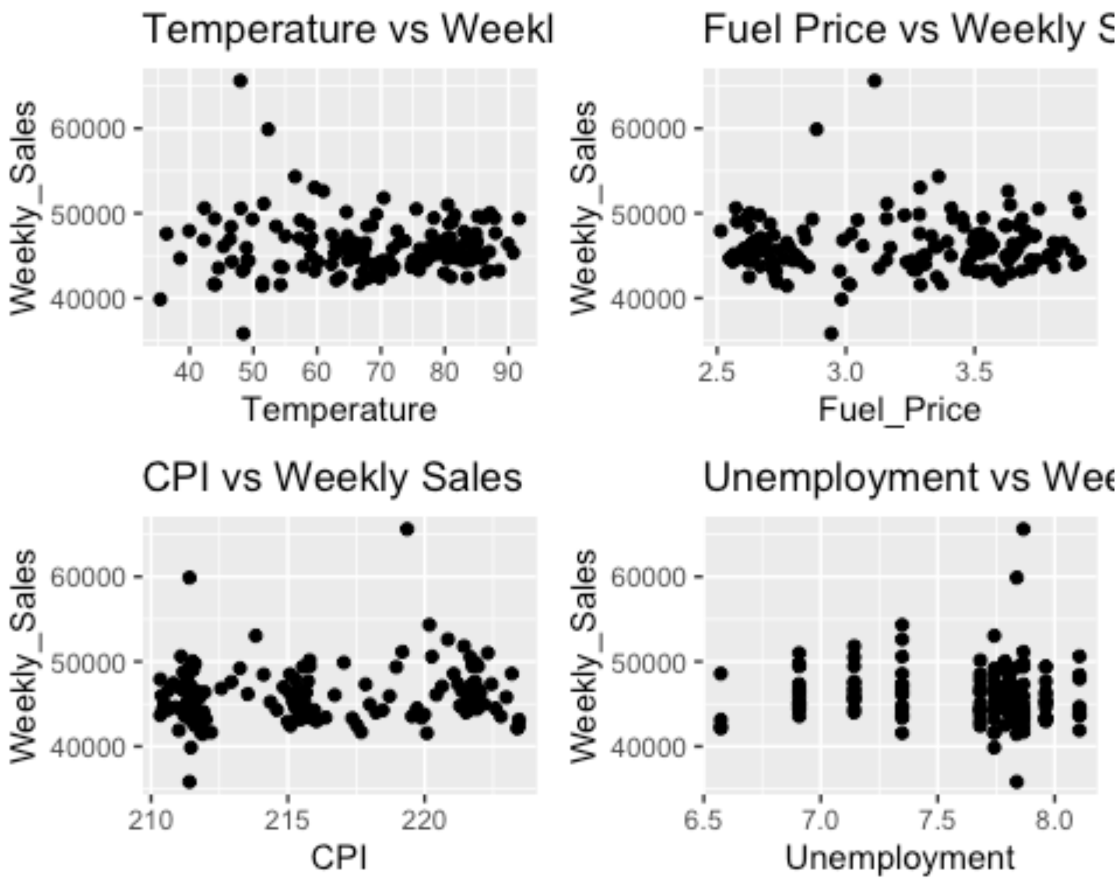


Scatter Plots of Each Departments Weekly_Sales vs. Unemployment/CPI/Temperature/Unemployment

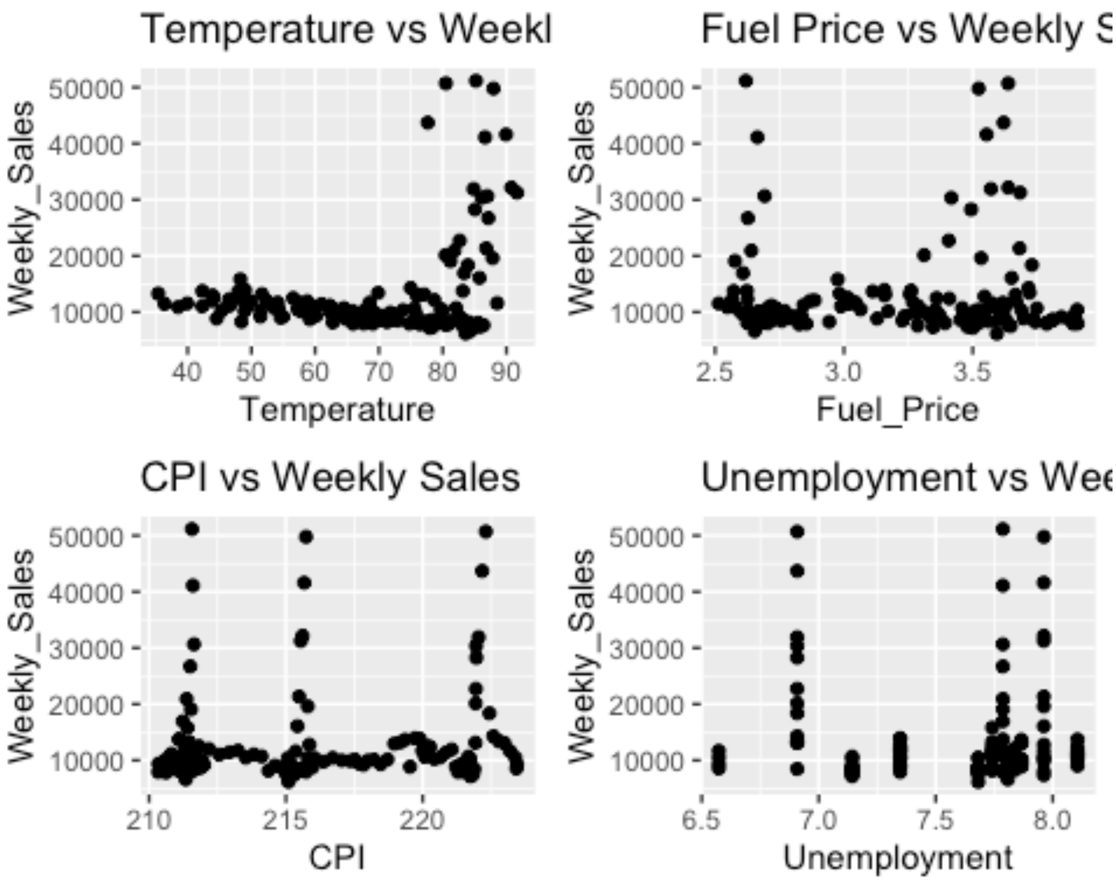
```
scatter_1.1.1 <- ggplot(store_1.1, aes(x = Temperature, y = Weekly_Sales)) +
  ggtitle("Temperature vs Weekly Sales") +
  geom_point()

scatter_1.1.2 <- ggplot(store_1.1, aes(x = Fuel_Price, y = Weekly_Sales)) +
  ggtitle("Fuel Price vs Weekly Sales") +
  geom_point()

scatter_1.1.3 <- ggplot(store_1.1, aes(x = CPI, y = Weekly_Sales)) +
  ggtitle("CPI vs Weekly Sales") +
  geom_point()

scatter_1.1.4 <- ggplot(store_1.1, aes(x = Unemployment, y = Weekly_Sales)) +
  ggtitle("Unemployment vs Weekly Sales") +
  geom_point()

grid.arrange(scatter_1.1.1, scatter_1.1.2, scatter_1.1.3, scatter_1.1.4)
```



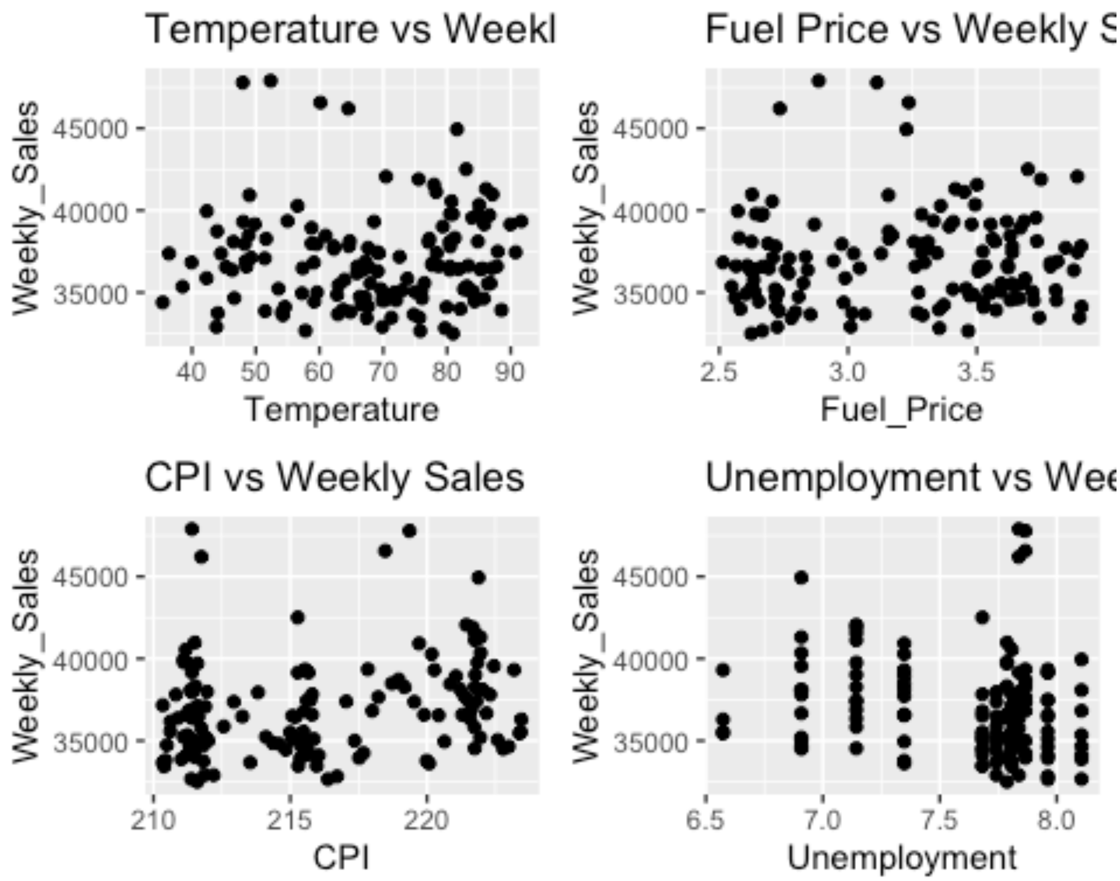

```
scatter_1.2.1 <- ggplot(store_1.2, aes(x = Temperature, y = Weekly_Sales)) +
  ggtitle("Temperature vs Weekly Sales") +
  geom_point()

scatter_1.2.2 <- ggplot(store_1.2, aes(x = Fuel_Price, y = Weekly_Sales)) +
  ggtitle("Fuel Price vs Weekly Sales") +
  geom_point()

scatter_1.2.3 <- ggplot(store_1.2, aes(x = CPI, y = Weekly_Sales)) +
  ggtitle("CPI vs Weekly Sales") +
  geom_point()

scatter_1.2.4 <- ggplot(store_1.2, aes(x = Unemployment, y = Weekly_Sales)) +
  ggtitle("Unemployment vs Weekly Sales") +
  geom_point()

grid.arrange(scatter_1.2.1, scatter_1.2.2, scatter_1.2.3, scatter_1.2.4)
```



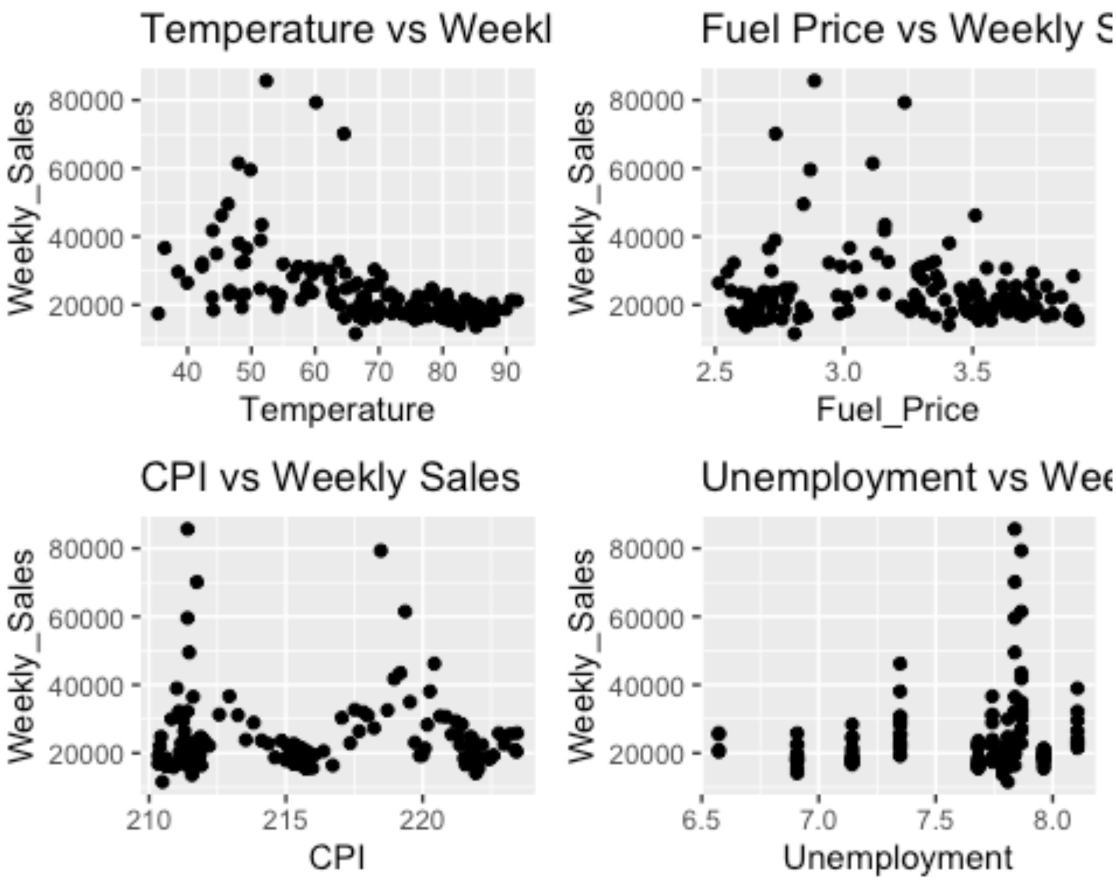

```
scatter_1.3.1 <- ggplot(store_1.3, aes(x = Temperature, y = Weekly_Sales)) +
  ggtitle("Temperature vs Weekly Sales") +
  geom_point()

scatter_1.3.2 <- ggplot(store_1.3, aes(x = Fuel_Price, y = Weekly_Sales)) +
  ggtitle("Fuel Price vs Weekly Sales") +
  geom_point()

scatter_1.3.3 <- ggplot(store_1.3, aes(x = CPI, y = Weekly_Sales)) +
  ggtitle("CPI vs Weekly Sales") +
  geom_point()

scatter_1.3.4 <- ggplot(store_1.3, aes(x = Unemployment, y = Weekly_Sales)) +
  ggtitle("Unemployment vs Weekly Sales") +
  geom_point()

grid.arrange(scatter_1.3.1, scatter_1.3.2, scatter_1.3.3, scatter_1.3.4)
```



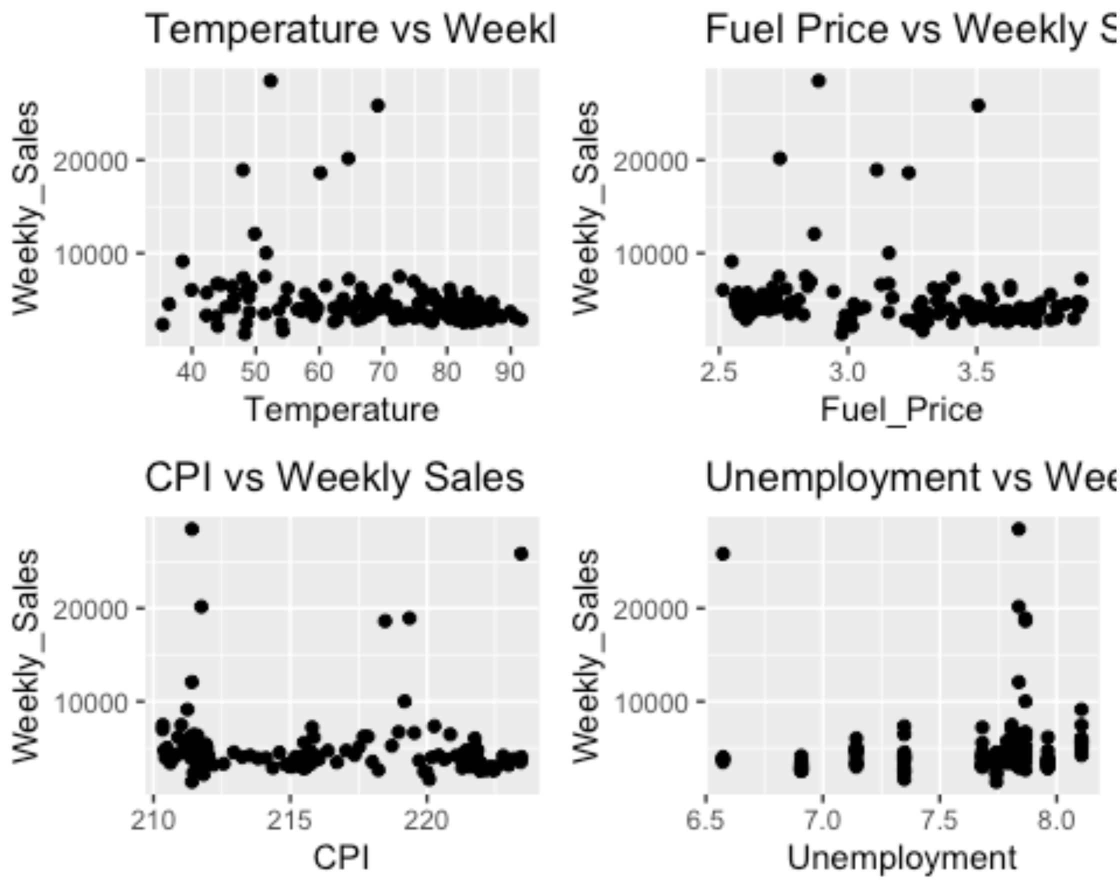

```
scatter_1.4.1 <- ggplot(store_1.4, aes(x = Temperature, y = Weekly_Sales)) +
  ggtitle("Temperature vs Weekly Sales") +
  geom_point()

scatter_1.4.2 <- ggplot(store_1.4, aes(x = Fuel_Price, y = Weekly_Sales)) +
  ggtitle("Fuel Price vs Weekly Sales") +
  geom_point()

scatter_1.4.3 <- ggplot(store_1.4, aes(x = CPI, y = Weekly_Sales)) +
  ggtitle("CPI vs Weekly Sales") +
  geom_point()

scatter_1.4.4 <- ggplot(store_1.4, aes(x = Unemployment, y = Weekly_Sales)) +
  ggtitle("Unemployment vs Weekly Sales") +
  geom_point()

grid.arrange(scatter_1.4.1, scatter_1.4.2, scatter_1.4.3, scatter_1.4.4)
```



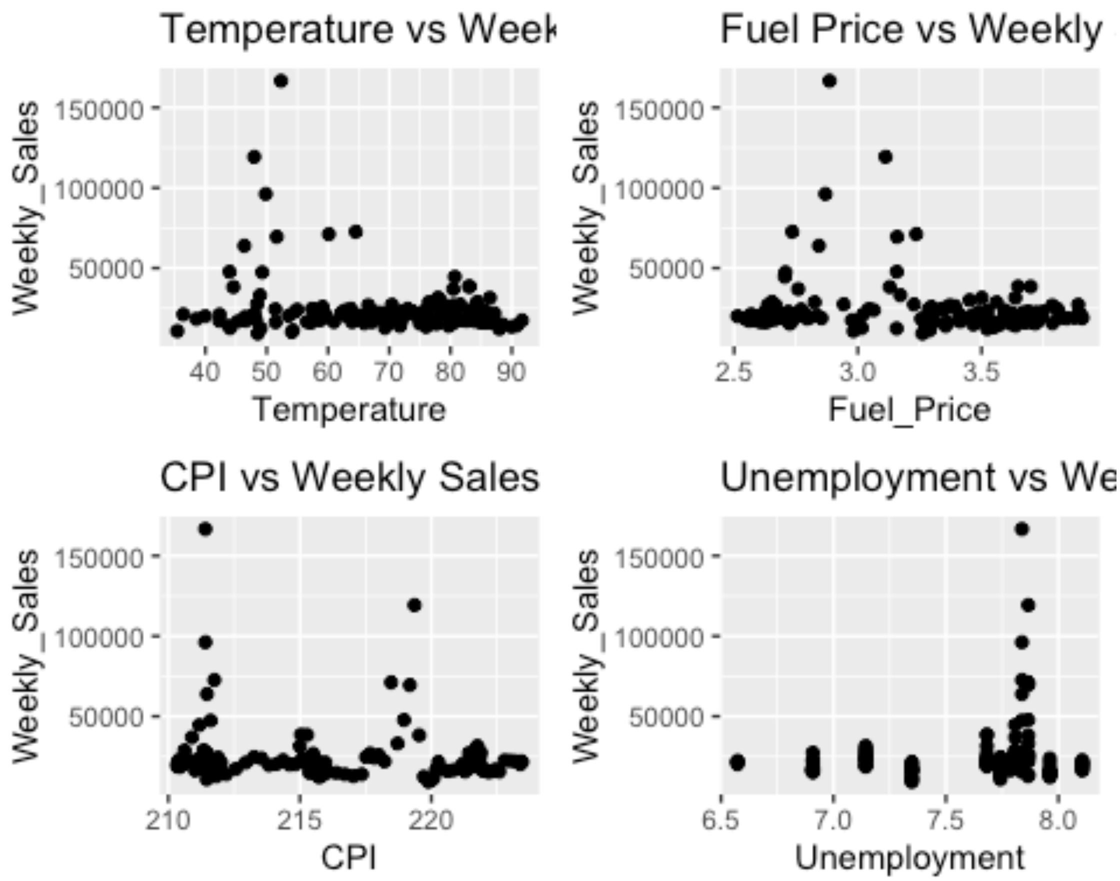

```r
scatter_1.5.1 <- ggplot(store_1.5, aes(x = Temperature, y = Weekly_Sales)) +
  ggtitle("Temperature vs Weekly Sales") +
  geom_point()

scatter_1.5.2 <- ggplot(store_1.5, aes(x = Fuel_Price, y = Weekly_Sales)) +
  ggtitle("Fuel Price vs Weekly Sales") +
  geom_point()

scatter_1.5.3 <- ggplot(store_1.5, aes(x = CPI, y = Weekly_Sales)) +
  ggtitle("CPI vs Weekly Sales") +
  geom_point()

scatter_1.5.4 <- ggplot(store_1.5, aes(x = Unemployment, y = Weekly_Sales)) +
  ggtitle("Unemployment vs Weekly Sales") +
  geom_point()

grid.arrange(scatter_1.5.1, scatter_1.5.2, scatter_1.5.3, scatter_1.5.4)
```



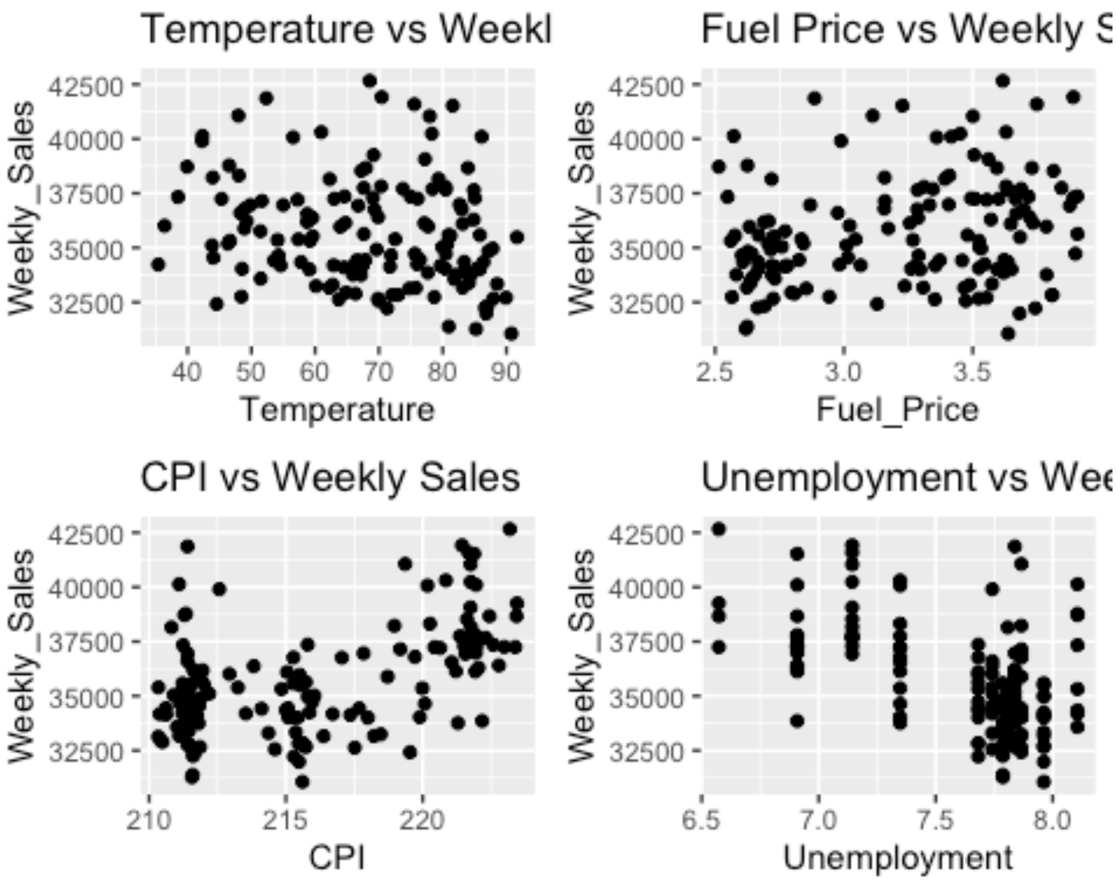

```
scatter_1.6.1 <- ggplot(store_1.6, aes(x = Temperature, y = Weekly_Sales)) +
  ggtitle("Temperature vs Weekly Sales") +
  geom_point()

scatter_1.6.2 <- ggplot(store_1.6, aes(x = Fuel_Price, y = Weekly_Sales)) +
  ggtitle("Fuel Price vs Weekly Sales") +
  geom_point()

scatter_1.6.3 <- ggplot(store_1.6, aes(x = CPI, y = Weekly_Sales)) +
  ggtitle("CPI vs Weekly Sales") +
  geom_point()

scatter_1.6.4 <- ggplot(store_1.6, aes(x = Unemployment, y = Weekly_Sales)) +
  ggtitle("Unemployment vs Weekly Sales") +
  geom_point()

grid.arrange(scatter_1.6.1, scatter_1.6.2, scatter_1.6.3, scatter_1.6.4)
```



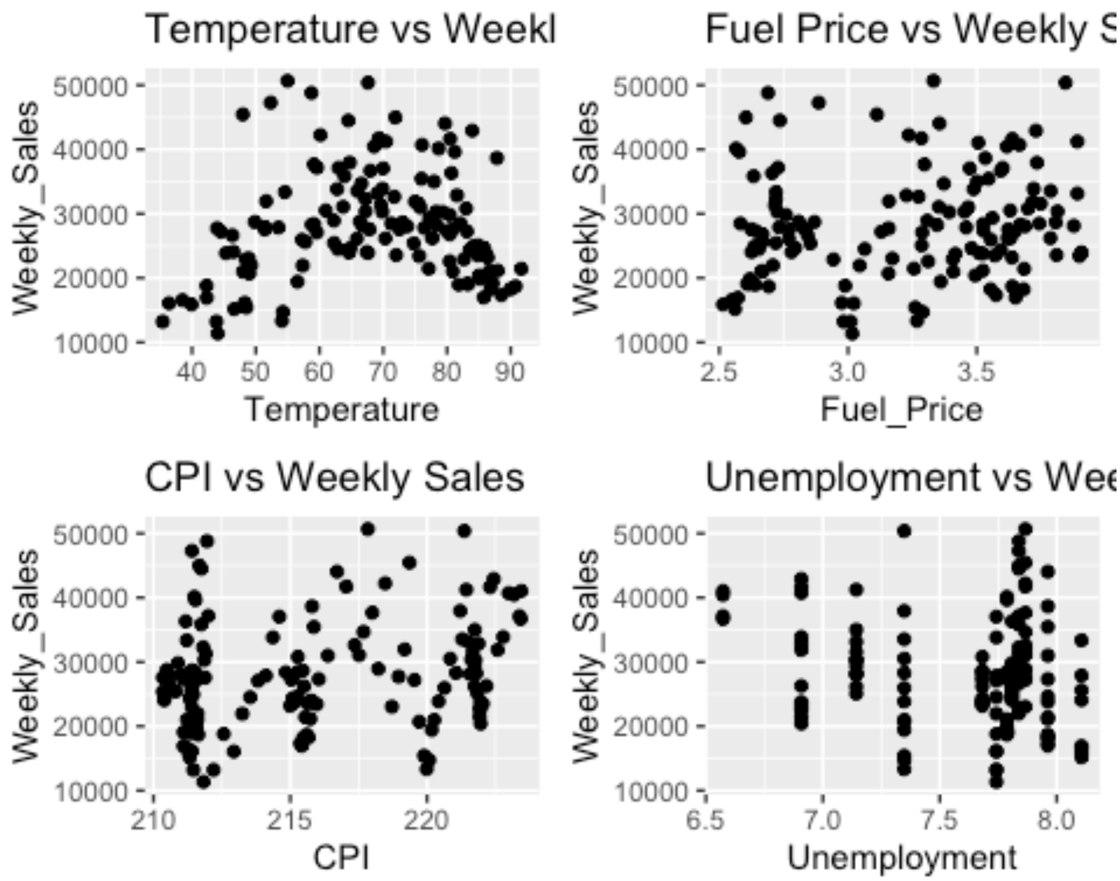

```r
scatter_1.7.1 <- ggplot(store_1.7, aes(x = Temperature, y = Weekly_Sales)) +
  ggtitle("Temperature vs Weekly Sales") +
  geom_point()

scatter_1.7.2 <- ggplot(store_1.7, aes(x = Fuel_Price, y = Weekly_Sales)) +
  ggtitle("Fuel Price vs Weekly Sales") +
  geom_point()

scatter_1.7.3 <- ggplot(store_1.7, aes(x = CPI, y = Weekly_Sales)) +
  ggtitle("CPI vs Weekly Sales") +
  geom_point()

scatter_1.7.4 <- ggplot(store_1.7, aes(x = Unemployment, y = Weekly_Sales)) +
  ggtitle("Unemployment vs Weekly Sales") +
  geom_point()

grid.arrange(scatter_1.7.1, scatter_1.7.2, scatter_1.7.3, scatter_1.7.4)
```



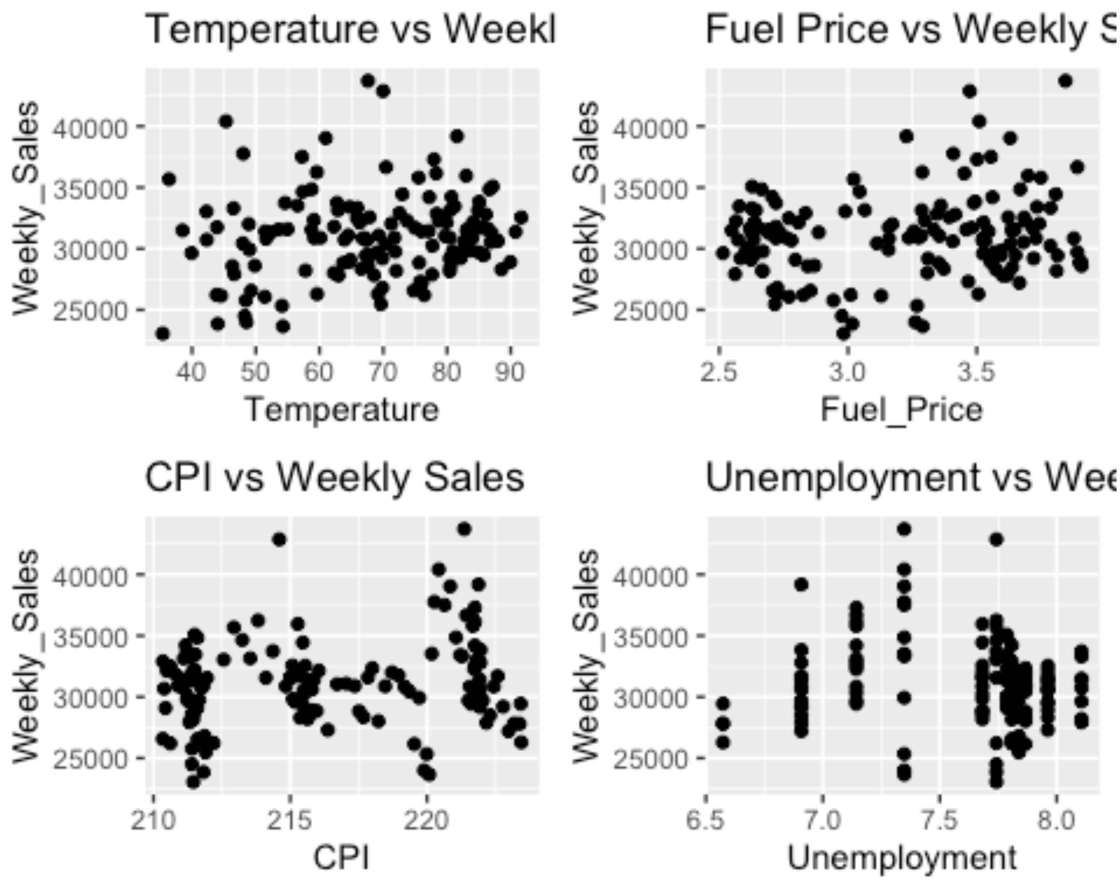

```
scatter_1.8.1 <- ggplot(store_1.8, aes(x = Temperature, y = Weekly_Sales)) +
  ggtitle("Temperature vs Weekly Sales") +
  geom_point()

scatter_1.8.2 <- ggplot(store_1.8, aes(x = Fuel_Price, y = Weekly_Sales)) +
  ggtitle("Fuel Price vs Weekly Sales") +
  geom_point()

scatter_1.8.3 <- ggplot(store_1.8, aes(x = CPI, y = Weekly_Sales)) +
  ggtitle("CPI vs Weekly Sales") +
  geom_point()

scatter_1.8.4 <- ggplot(store_1.8, aes(x = Unemployment, y = Weekly_Sales)) +
  ggtitle("Unemployment vs Weekly Sales") +
  geom_point()

grid.arrange(scatter_1.8.1, scatter_1.8.2, scatter_1.8.3, scatter_1.8.4)
```



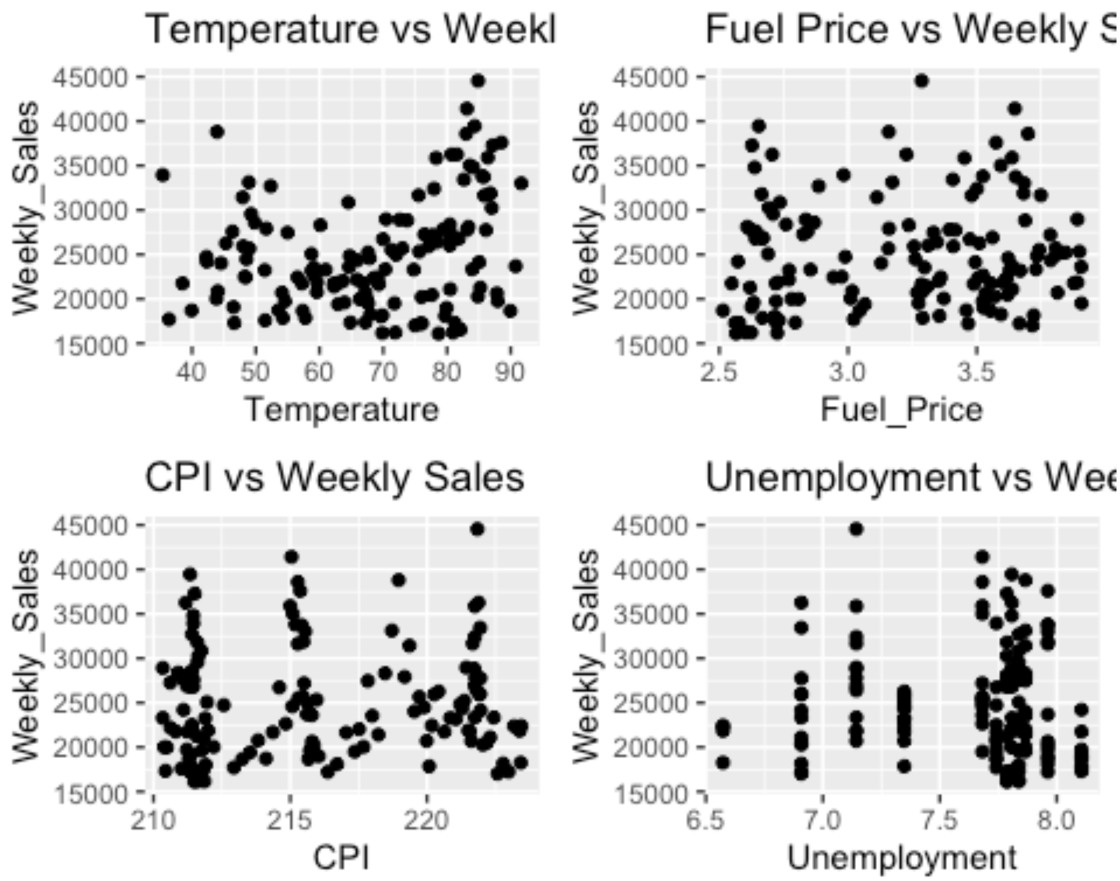

```
scatter_1.9.1 <- ggplot(store_1.9, aes(x = Temperature, y = Weekly_Sales)) +
  ggtitle("Temperature vs Weekly Sales") +
  geom_point()

scatter_1.9.2 <- ggplot(store_1.9, aes(x = Fuel_Price, y = Weekly_Sales)) +
  ggtitle("Fuel Price vs Weekly Sales") +
  geom_point()

scatter_1.9.3 <- ggplot(store_1.9, aes(x = CPI, y = Weekly_Sales)) +
  ggtitle("CPI vs Weekly Sales") +
  geom_point()

scatter_1.9.4 <- ggplot(store_1.9, aes(x = Unemployment, y = Weekly_Sales)) +
  ggtitle("Unemployment vs Weekly Sales") +
  geom_point()

grid.arrange(scatter_1.9.1, scatter_1.9.2, scatter_1.9.3, scatter_1.9.4)
```



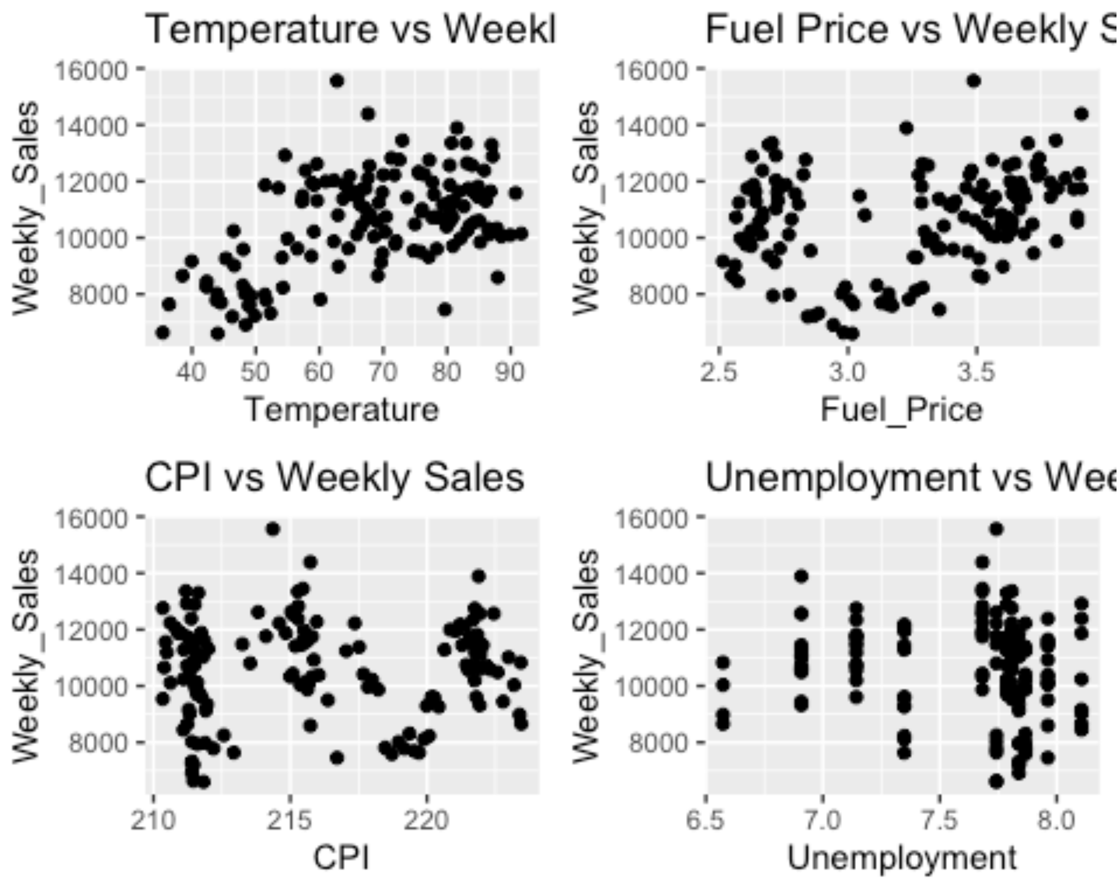

```
scatter_1.10.1 <- ggplot(store_1.10, aes(x = Temperature, y = Weekly_Sales))
+
  ggtitle("Temperature vs Weekly Sales") +
  geom_point()

scatter_1.10.2 <- ggplot(store_1.10, aes(x = Fuel_Price, y = Weekly_Sales)) +
  ggtitle("Fuel Price vs Weekly Sales") +
  geom_point()

scatter_1.10.3 <- ggplot(store_1.10, aes(x = CPI, y = Weekly_Sales)) +
  ggtitle("CPI vs Weekly Sales") +
  geom_point()

scatter_1.10.4 <- ggplot(store_1.10, aes(x = Unemployment, y = Weekly_Sales))
+
  ggtitle("Unemployment vs Weekly Sales") +
  geom_point()

grid.arrange(scatter_1.10.1, scatter_1.10.2, scatter_1.10.3, scatter_1.10.4)
```



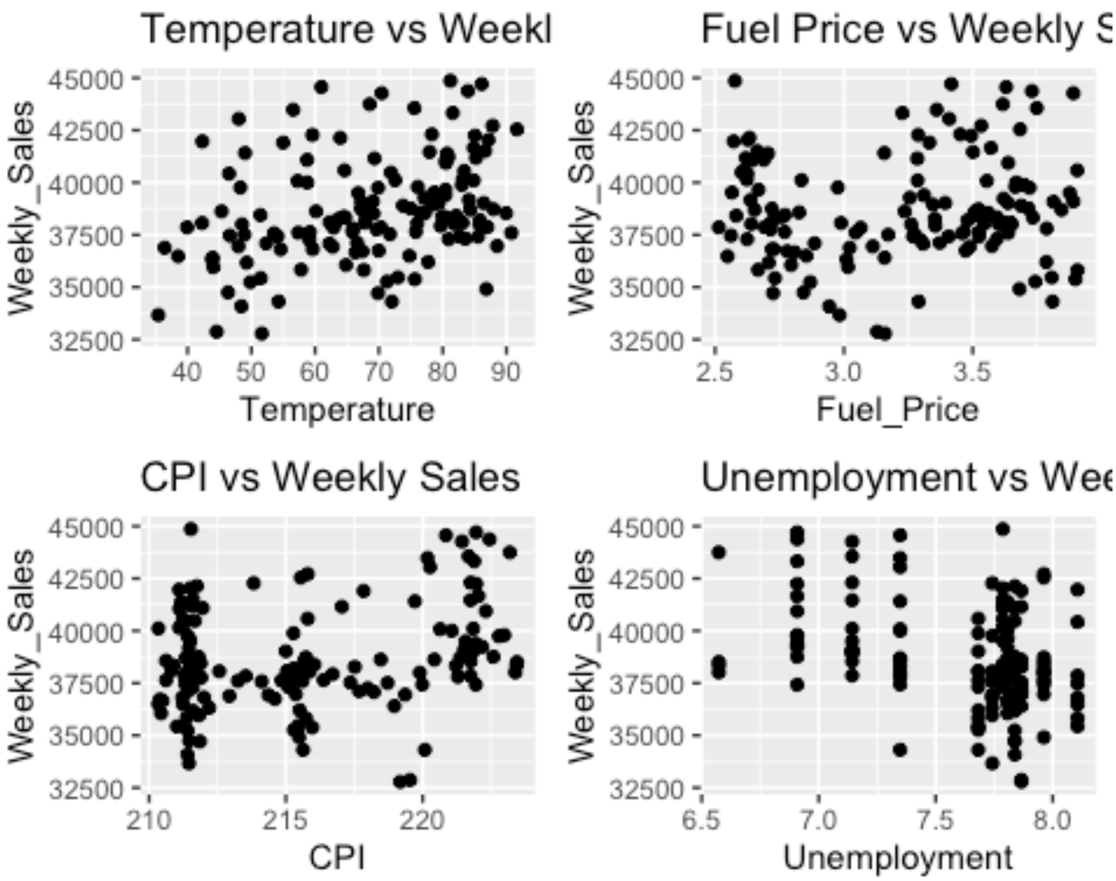

```
scatter_1.11.1 <- ggplot(store_1.11, aes(x = Temperature, y = Weekly_Sales))
+
  ggtitle("Temperature vs Weekly Sales") +
  geom_point()

scatter_1.11.2 <- ggplot(store_1.11, aes(x = Fuel_Price, y = Weekly_Sales)) +
  ggtitle("Fuel Price vs Weekly Sales") +
  geom_point()

scatter_1.11.3 <- ggplot(store_1.11, aes(x = CPI, y = Weekly_Sales)) +
  ggtitle("CPI vs Weekly Sales") +
  geom_point()

scatter_1.11.4 <- ggplot(store_1.11, aes(x = Unemployment, y = Weekly_Sales))
+
  ggtitle("Unemployment vs Weekly Sales") +
  geom_point()

grid.arrange(scatter_1.11.1, scatter_1.11.2, scatter_1.11.3, scatter_1.11.4)
```




```
scatter_1.12.1 <- ggplot(store_1.12, aes(x = Temperature, y = Weekly_Sales))
+
  ggtitle("Temperature vs Weekly Sales") +
  geom_point()

scatter_1.12.2 <- ggplot(store_1.12, aes(x = Fuel_Price, y = Weekly_Sales)) +
  ggtitle("Fuel Price vs Weekly Sales") +
  geom_point()

scatter_1.12.3 <- ggplot(store_1.12, aes(x = CPI, y = Weekly_Sales)) +
  ggtitle("CPI vs Weekly Sales") +
  geom_point()

scatter_1.12.4 <- ggplot(store_1.12, aes(x = Unemployment, y = Weekly_Sales))
+
  ggtitle("Unemployment vs Weekly Sales") +
  geom_point()

grid.arrange(scatter_1.12.1, scatter_1.12.2, scatter_1.12.3, scatter_1.12.4)
```

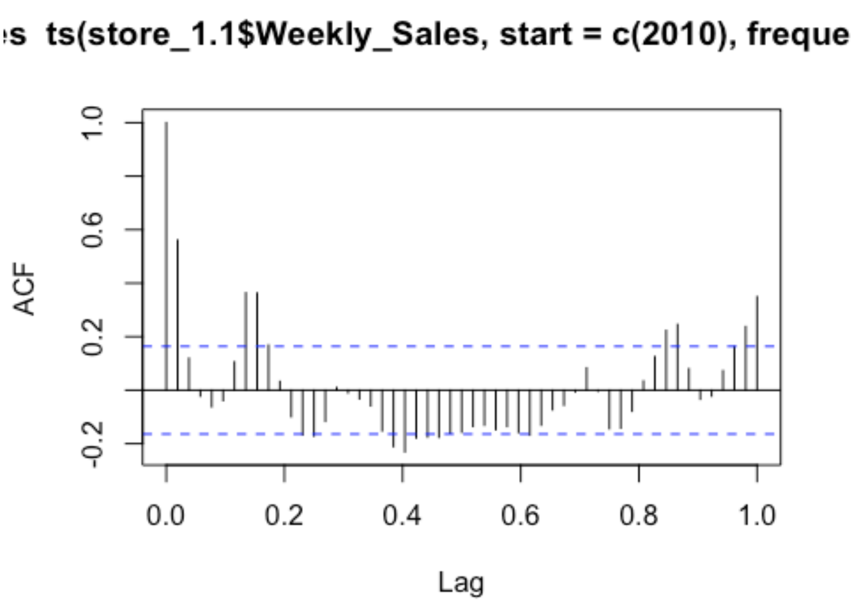



```
scatter_1.13.1 <- ggplot(store_1.13, aes(x = Temperature, y = Weekly_Sales))
+
  ggtitle("Temperature vs Weekly Sales") +
  geom_point()

scatter_1.13.2 <- ggplot(store_1.13, aes(x = Fuel_Price, y = Weekly_Sales)) +
  ggtitle("Fuel Price vs Weekly Sales") +
  geom_point()

scatter_1.13.3 <- ggplot(store_1.13, aes(x = CPI, y = Weekly_Sales)) +
  ggtitle("CPI vs Weekly Sales") +
  geom_point()

scatter_1.13.4 <- ggplot(store_1.13, aes(x = Unemployment, y = Weekly_Sales))
+
  ggtitle("Unemployment vs Weekly Sales") +
  geom_point()

grid.arrange(scatter_1.13.1, scatter_1.13.2, scatter_1.13.3, scatter_1.13.4)
```

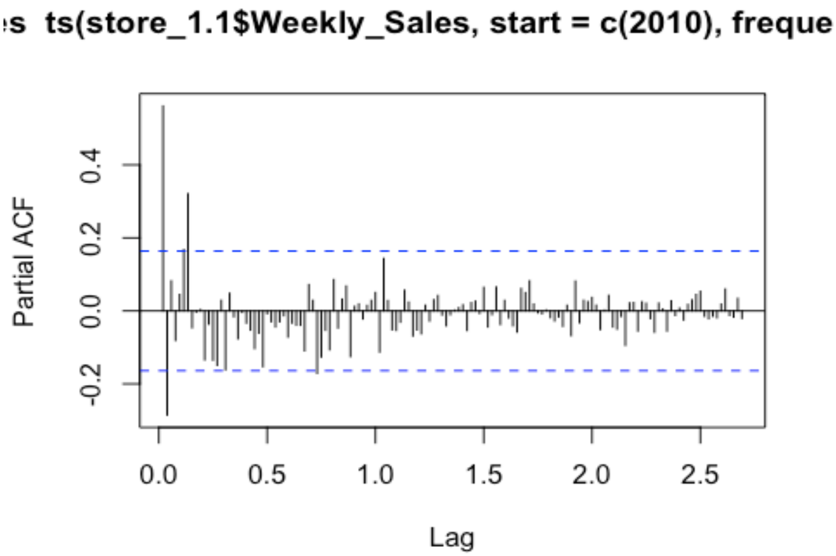


Autocorrelation Plots of Weekly_Sales

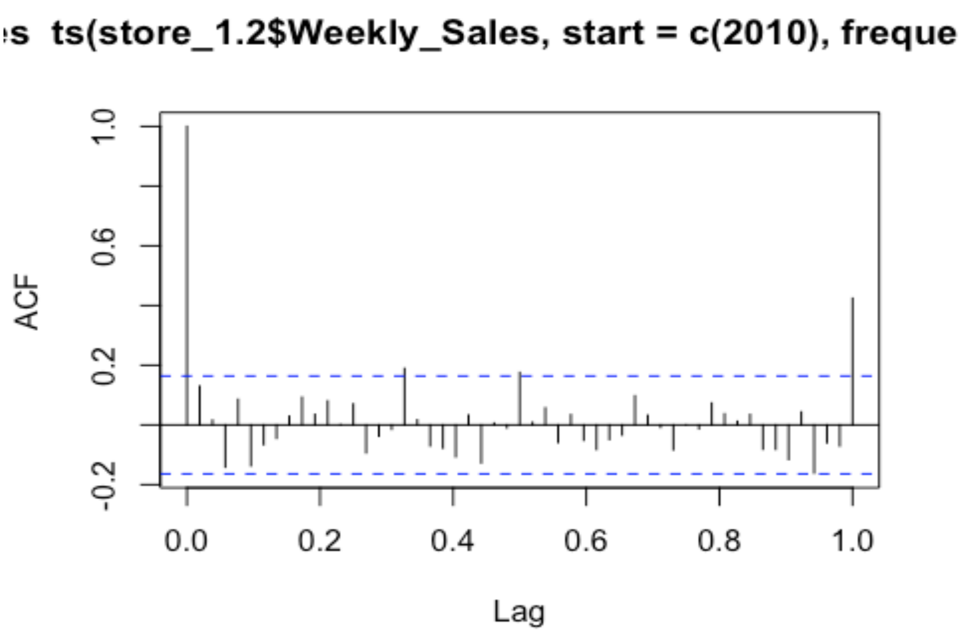
```r
# department 1
acf(ts(store_1.1$Weekly_Sales, start = c(2010), frequency = 52), lag = 52)
```



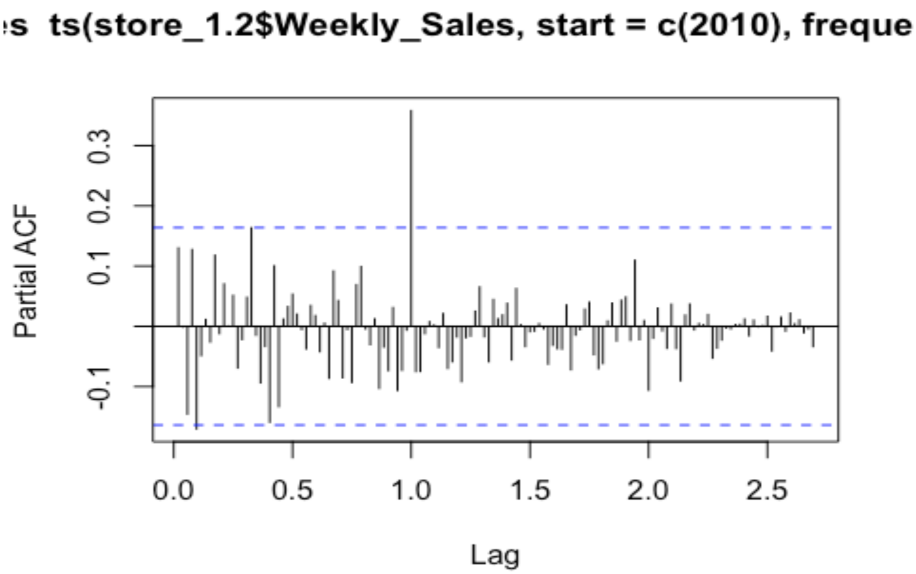
```
pacf(ts(store_1.1$Weekly_Sales, start = c(2010), frequency = 52), lag.max = 1
40)
```



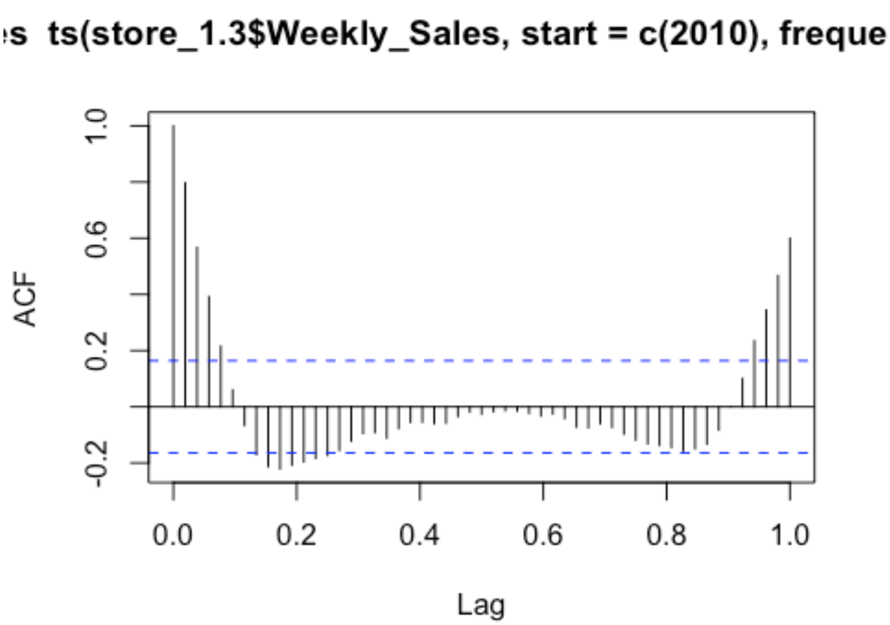

```
#department 2
acf(ts(store_1.2$Weekly_Sales, start = c(2010), frequency = 52), lag = 52)
```



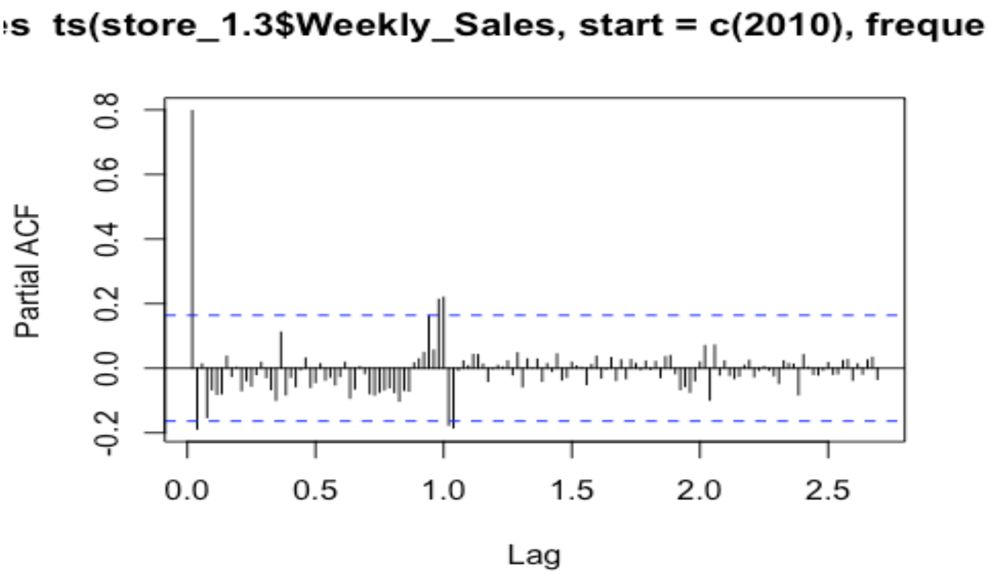

```
pacf(ts(store_1.2$Weekly_Sales, start = c(2010), frequency = 52), lag.max = 1
40)
```



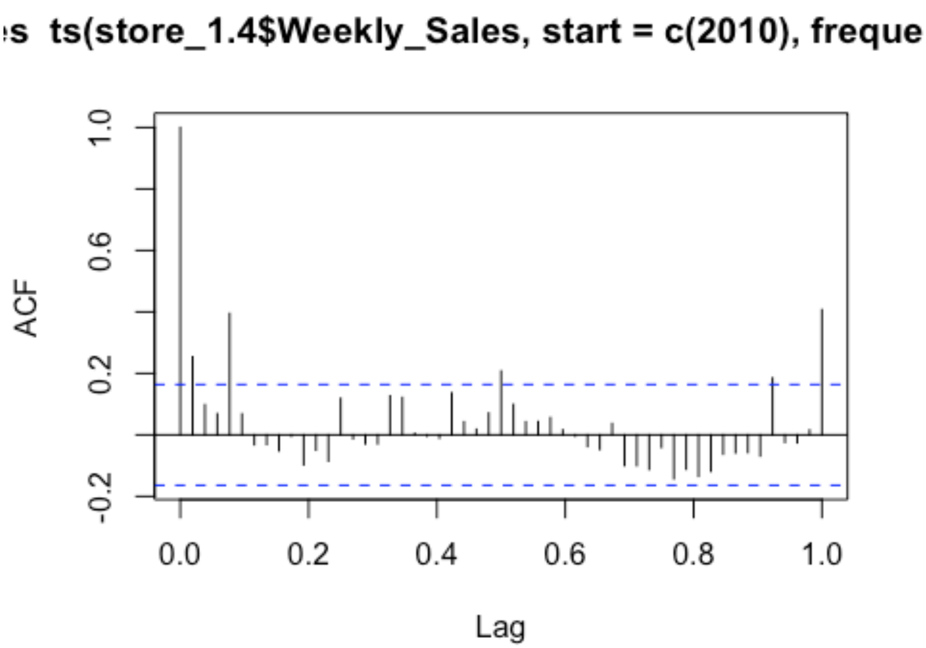

```r
# department 3
acf(ts(store_1.3$Weekly_Sales, start = c(2010), frequency = 52), lag = 52)
```



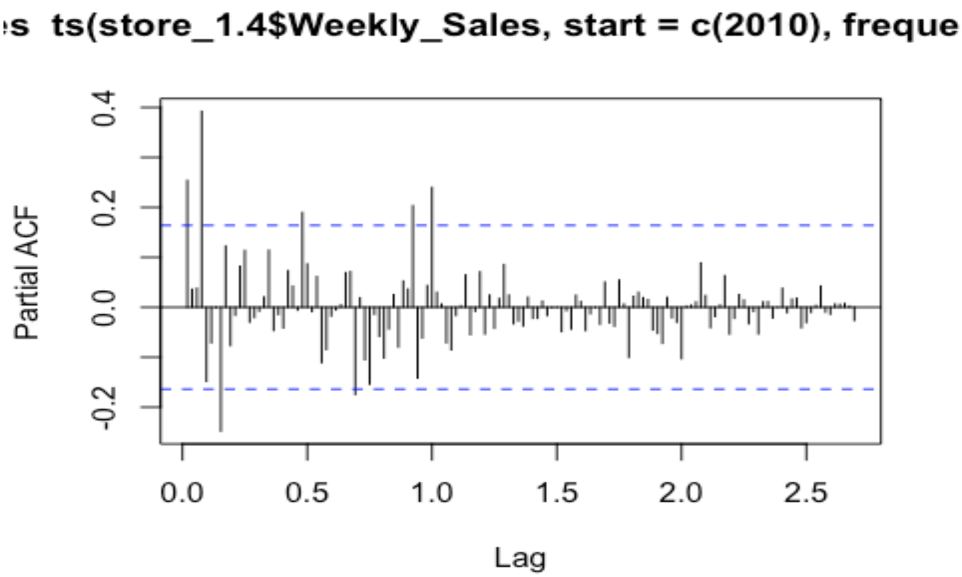

```r
pacf(ts(store_1.3$Weekly_Sales, start = c(2010), frequency = 52), lag.max = 1
40)
```



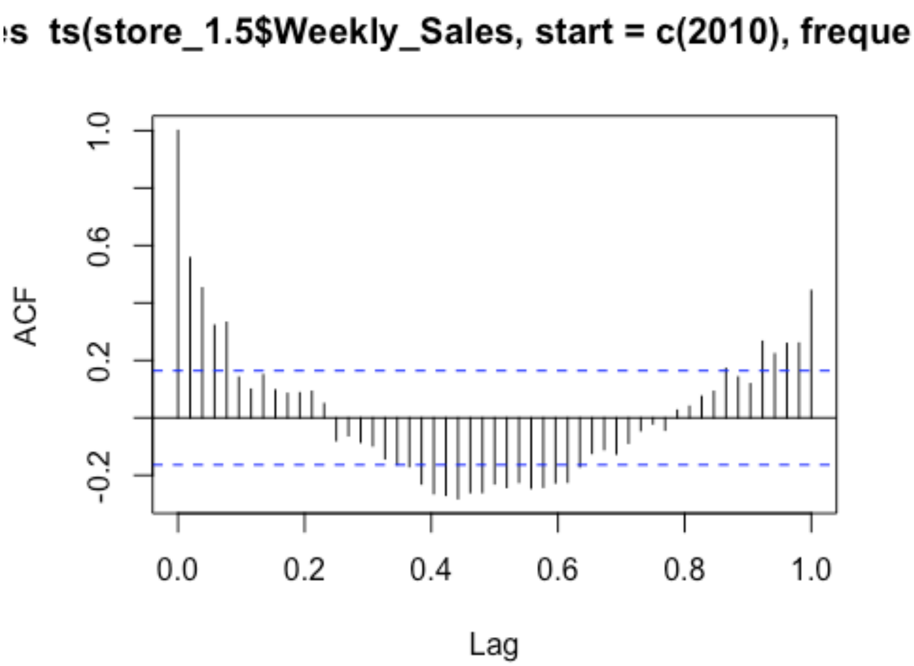

```r
# department 4
acf(ts(store_1.4$Weekly_Sales, start = c(2010), frequency = 52), lag = 52)
```



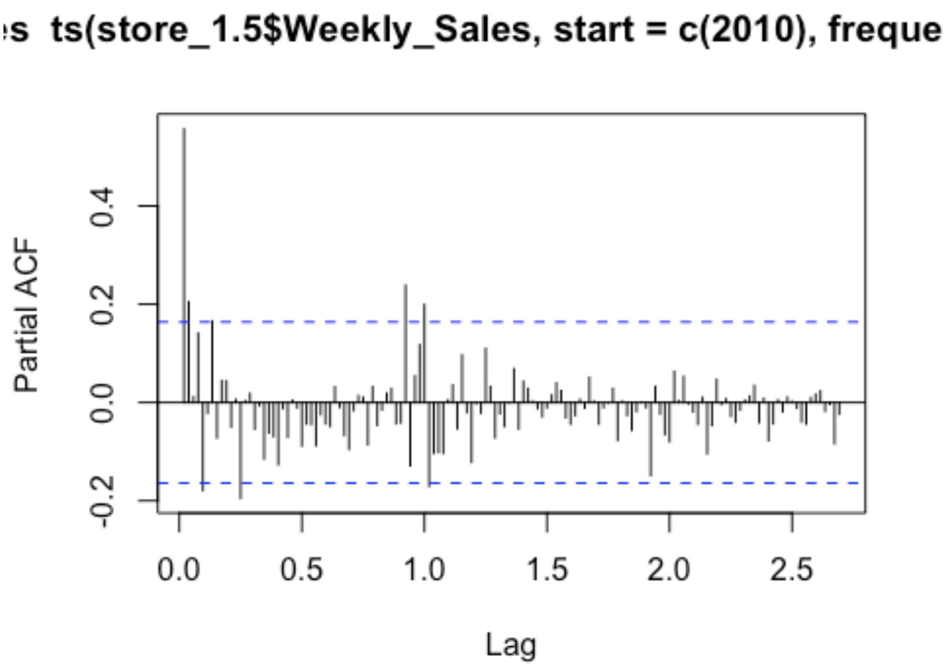

```r
pacf(ts(store_1.4$Weekly_Sales, start = c(2010), frequency = 52), lag.max = 1
40)
```



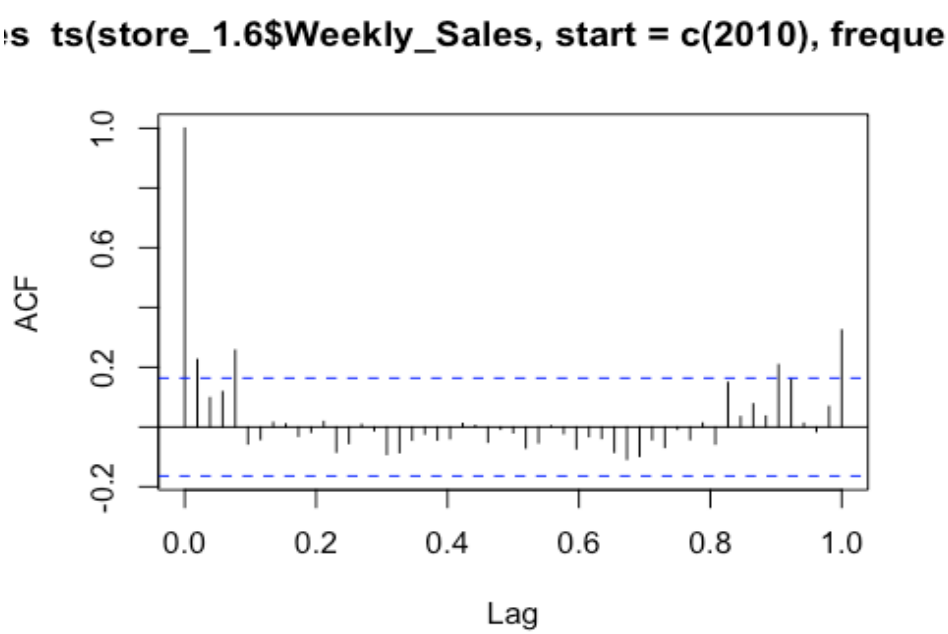

```r
# department 5
acf(ts(store_1.5$Weekly_Sales, start = c(2010), frequency = 52), lag =52)
```



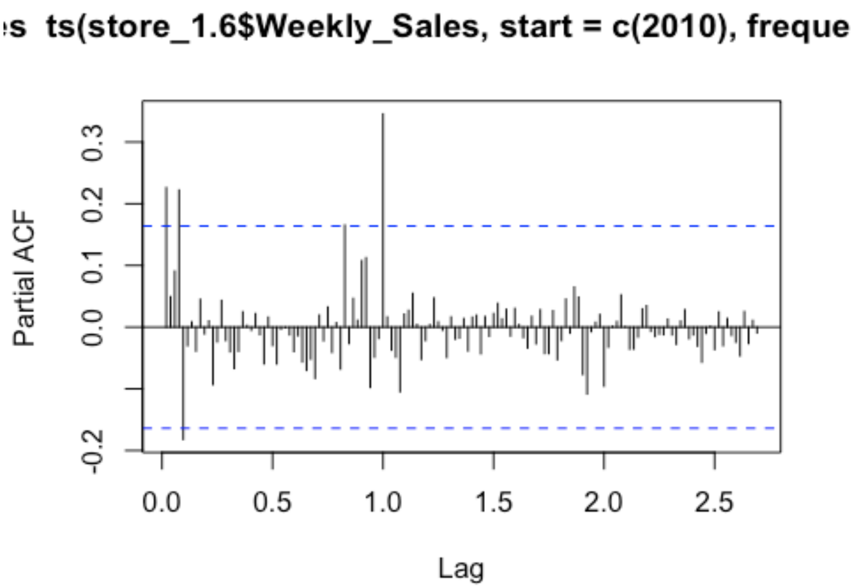

```r
pacf(ts(store_1.5$Weekly_Sales, start = c(2010), frequency = 52), lag.max = 1
40)
```



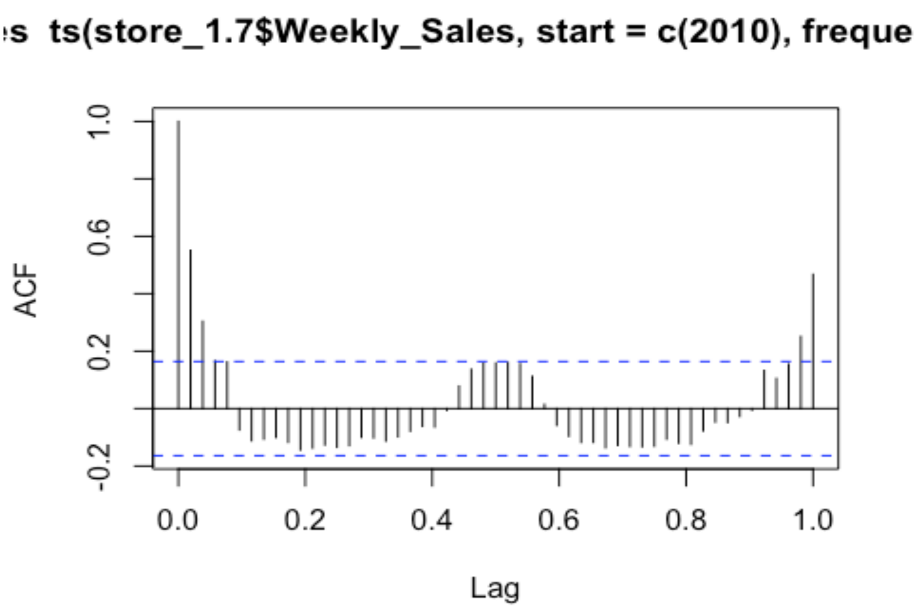

```r
# department 6
acf(ts(store_1.6$Weekly_Sales, start = c(2010), frequency = 52), lag = 52)
```



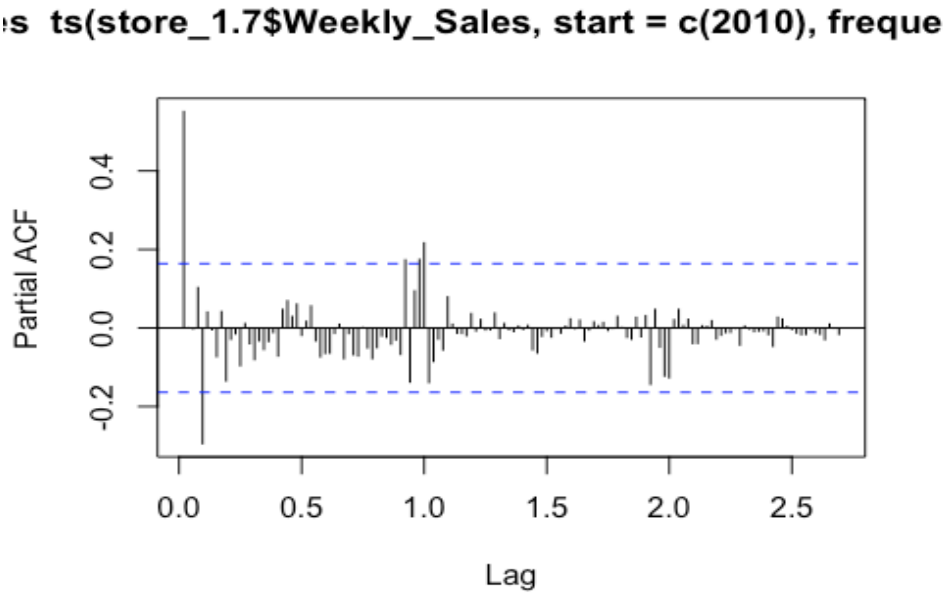

```r
pacf(ts(store_1.6$Weekly_Sales, start = c(2010), frequency = 52), lag.max = 1
40)
```



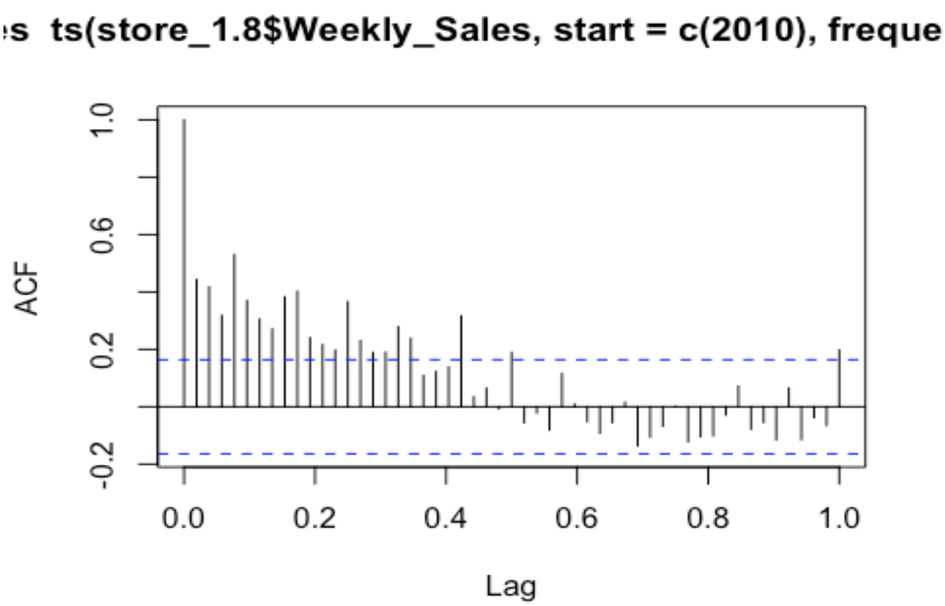

```
# department 7
acf(ts(store_1.7$Weekly_Sales, start = c(2010), frequency = 52), lag =52)
```



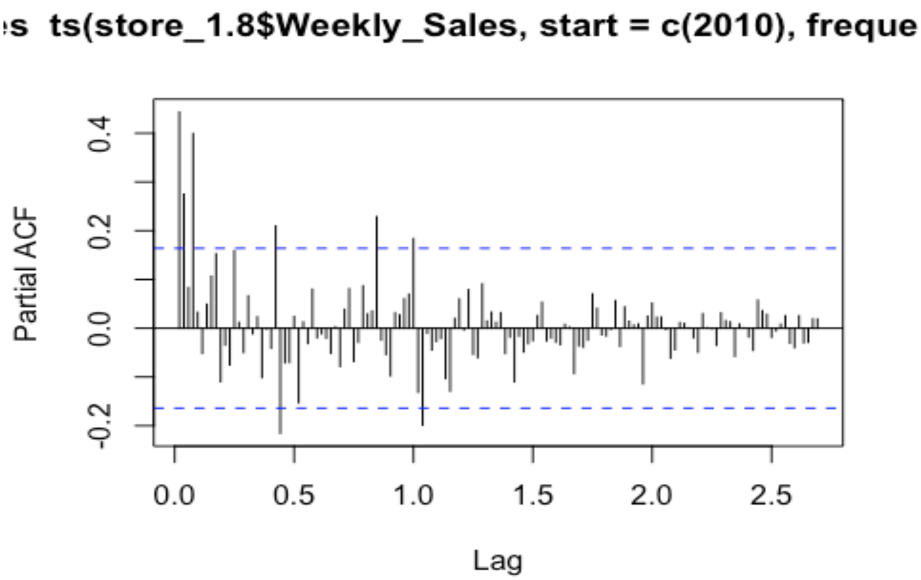

```
pacf(ts(store_1.7$Weekly_Sales, start = c(2010), frequency = 52), lag.max = 1
40)
```



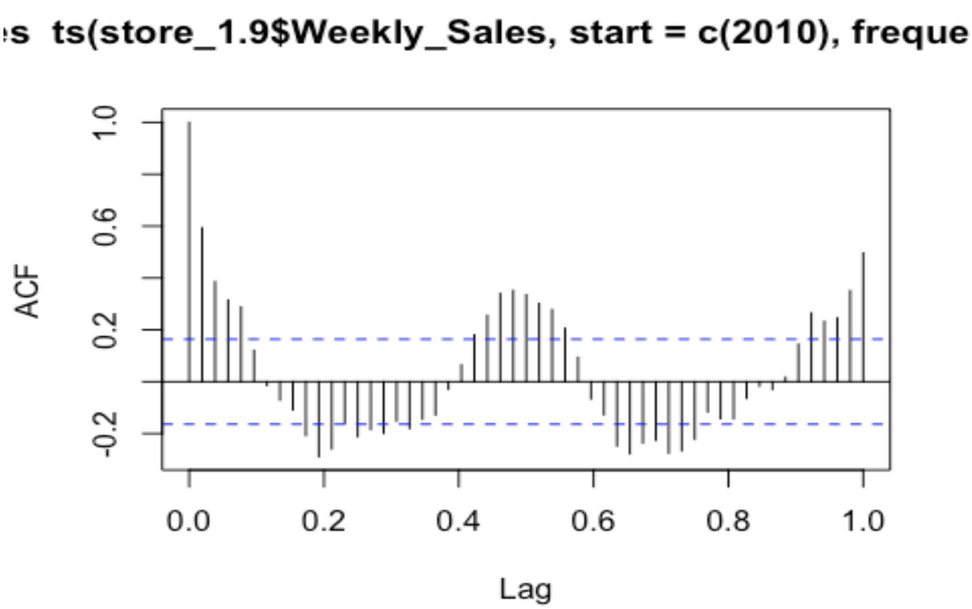

```r
# department 8
acf(ts(store_1.8$Weekly_Sales, start = c(2010), frequency = 52), lag = 52)
```



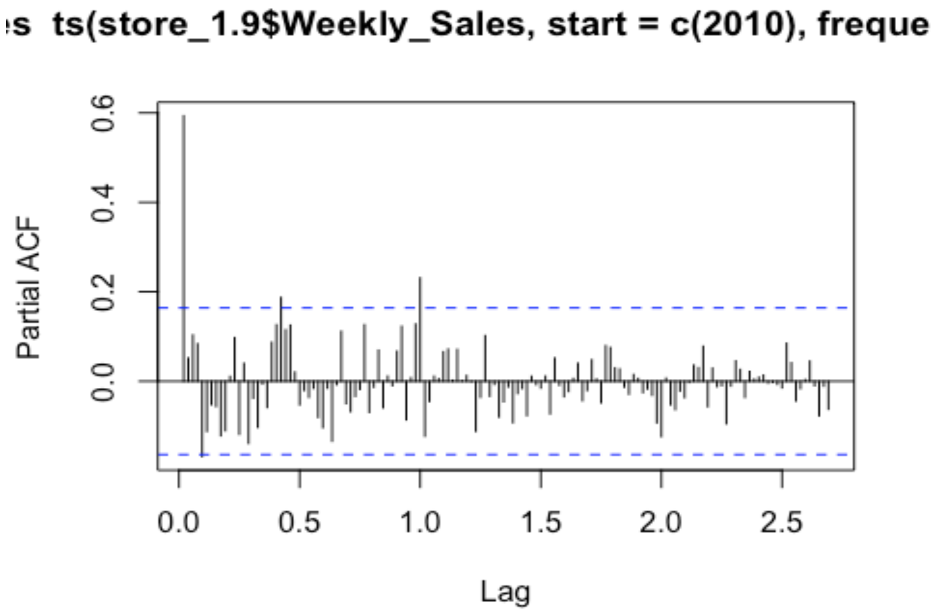

```r
pacf(ts(store_1.8$Weekly_Sales, start = c(2010), frequency = 52), lag.max = 1
40)
```



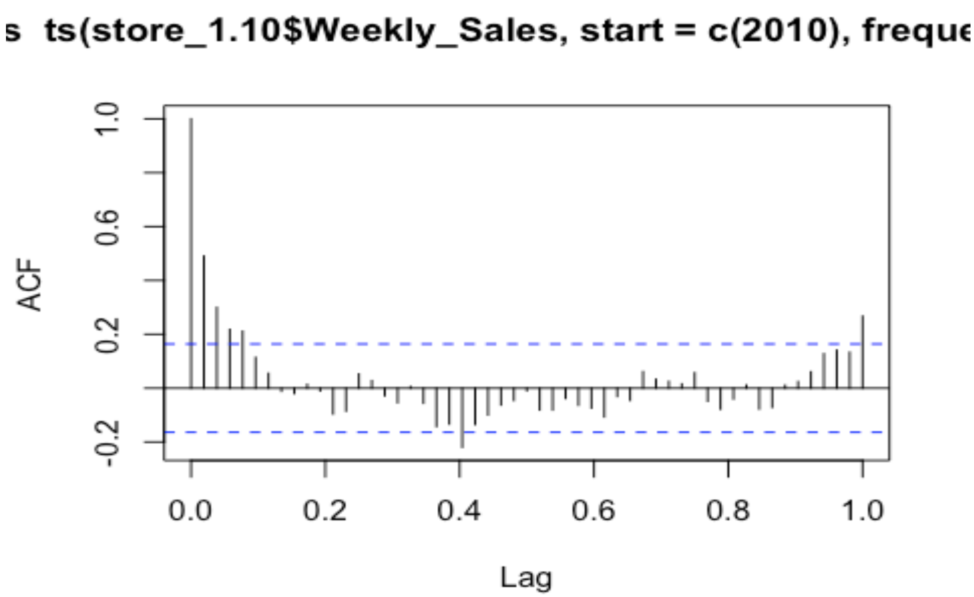

```
# department 9
acf(ts(store_1.9$Weekly_Sales, start = c(2010), frequency = 52), lag =52)
```



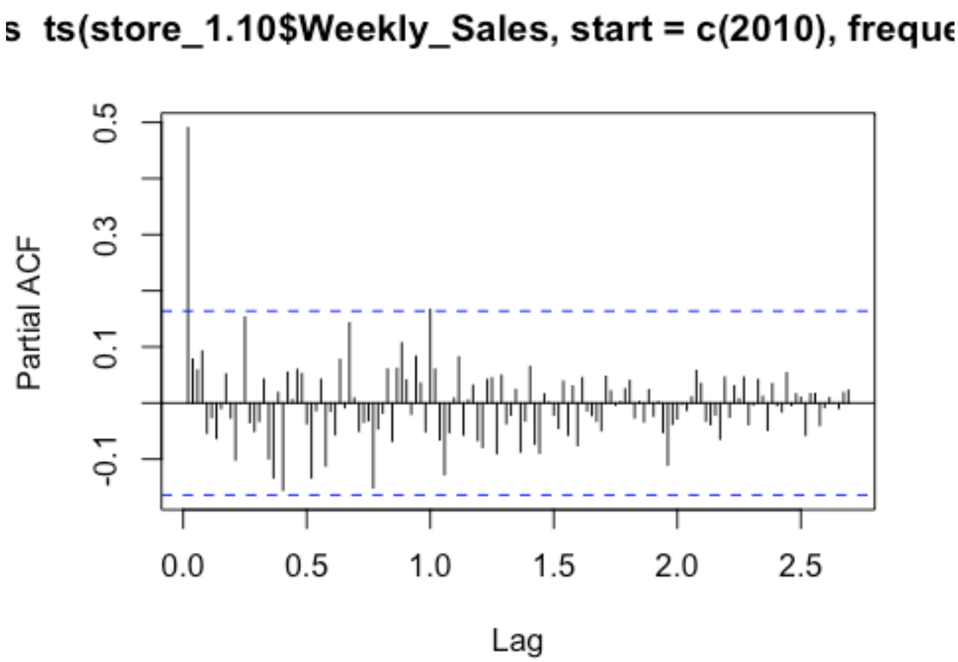

```
pacf(ts(store_1.9$Weekly_Sales, start = c(2010), frequency = 52), lag.max = 1
40)
```



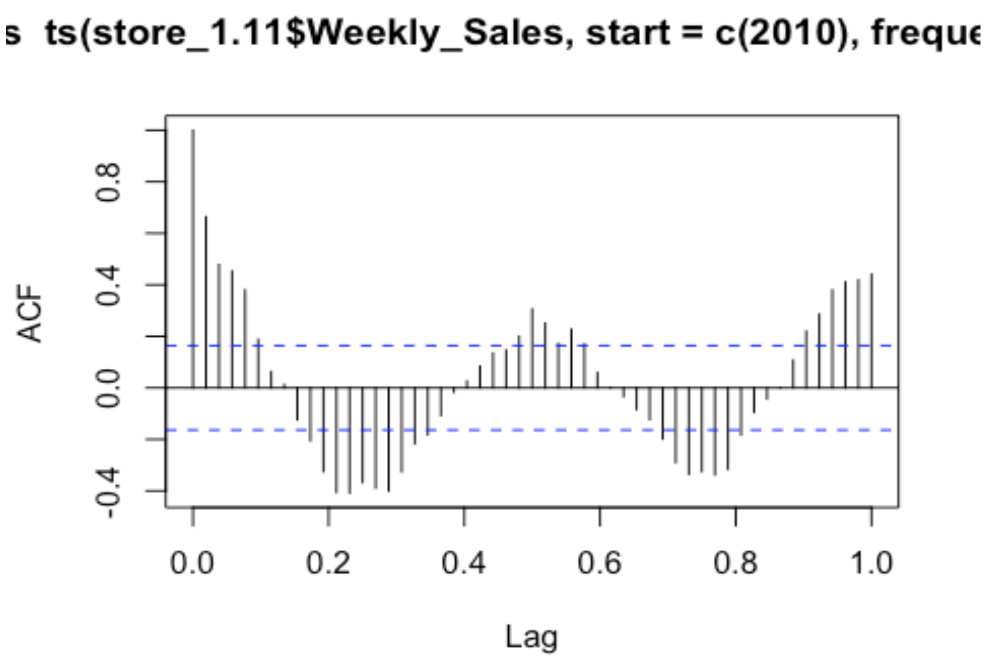

```
# department 10
acf(ts(store_1.10$Weekly_Sales, start = c(2010), frequency = 52), lag = 52)
```



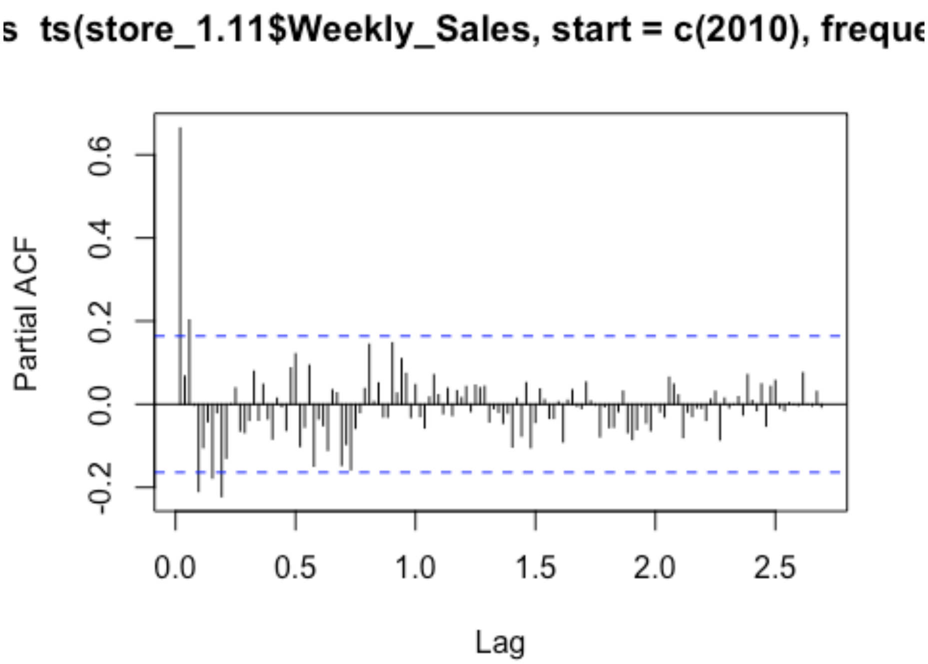

```
pacf(ts(store_1.10$Weekly_Sales, start = c(2010), frequency = 52), lag.max =
140)
```



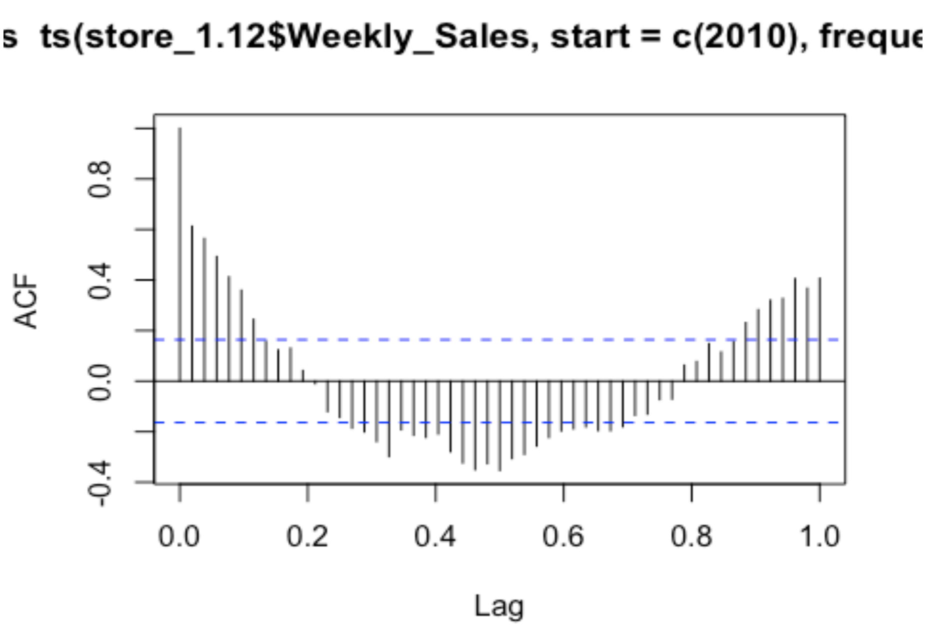

```r
# department 11
acf(ts(store_1.11$Weekly_Sales, start = c(2010), frequency = 52), lag = 52)
```



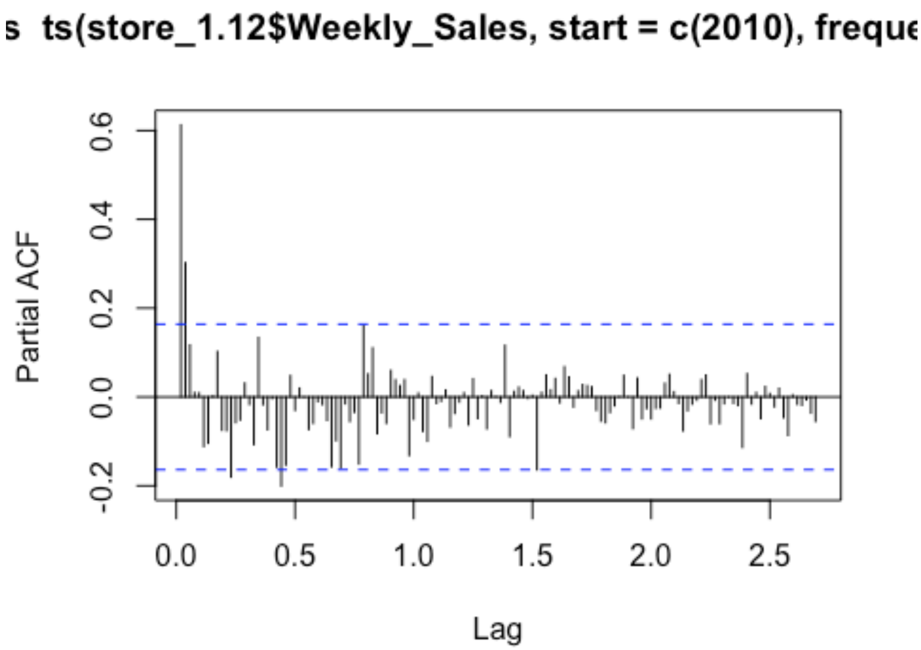

```r
pacf(ts(store_1.11$Weekly_Sales, start = c(2010), frequency = 52), lag.max =
140)
```



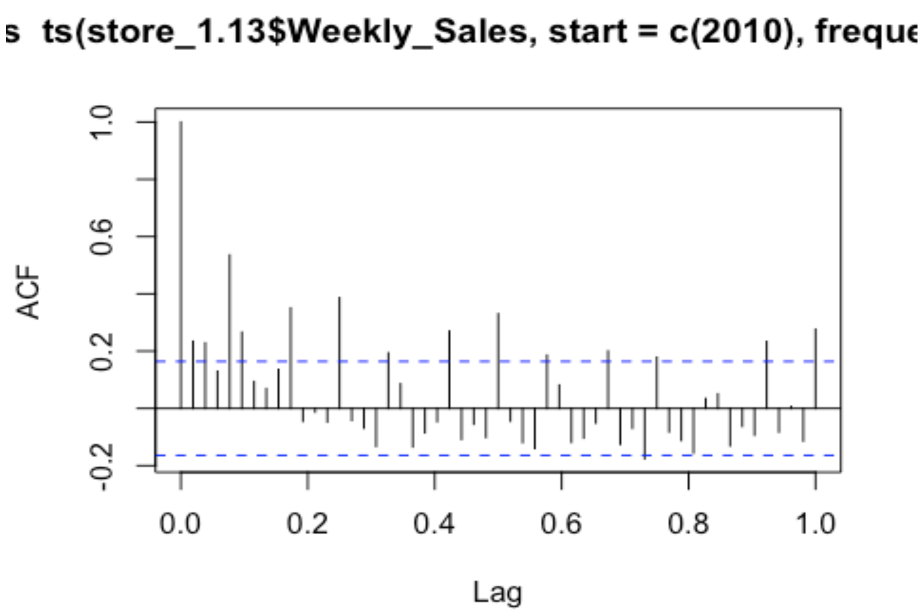

```r
# department 12
acf(ts(store_1.12$Weekly_Sales, start = c(2010), frequency = 52), lag = 52)
```



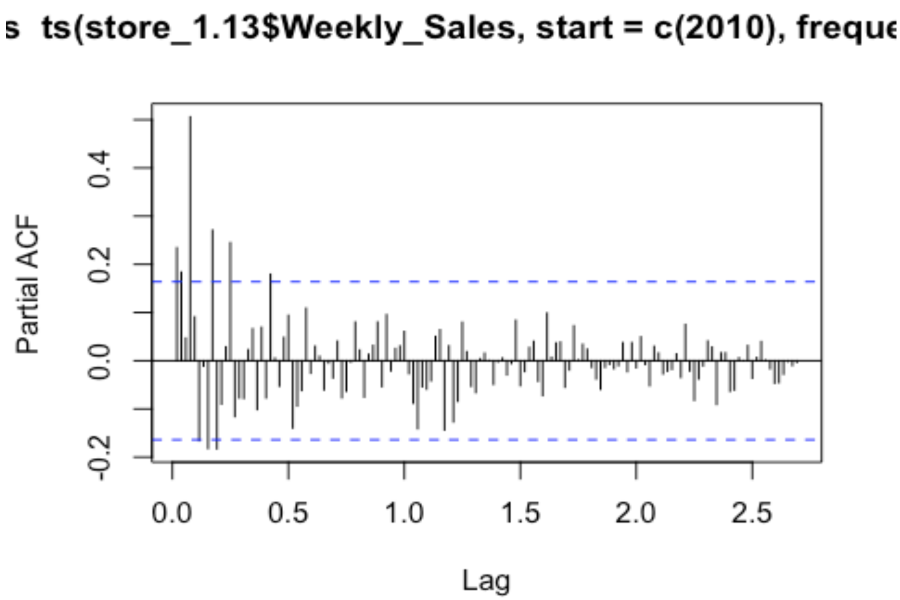

```r
pacf(ts(store_1.12$Weekly_Sales, start = c(2010), frequency = 52), lag.max =
140)
```




```
# department 13
acf(ts(store_1.13$Weekly_Sales, start = c(2010), frequency = 52), lag = 52)
```




```
pacf(ts(store_1.13$Weekly_Sales, start = c(2010), frequency = 52), lag.max =
140)
```



Data Cleaning:

Identifying and replacing outliers

```
# using tsclean function, which identifies and replaced outliers
store_1.1_clean <- tsclean(store_1.1_ts)
store_1.2_clean <- tsclean(store_1.2_ts)
store_1.3_clean <- tsclean(store_1.3_ts)
store_1.4_clean <- tsclean(store_1.4_ts)
store_1.5_clean <- tsclean(store_1.5_ts)
store_1.6_clean <- tsclean(store_1.6_ts)
store_1.7_clean <- tsclean(store_1.7_ts)
store_1.8_clean <- tsclean(store_1.8_ts)
store_1.9_clean <- tsclean(store_1.9_ts)
store_1.10_clean <- tsclean(store_1.10_ts)
store_1.11_clean <- tsclean(store_1.11_ts)
store_1.12_clean <- tsclean(store_1.12_ts)
store_1.13_clean <- tsclean(store_1.13_ts)
```

Modeling:

Partitioning Data

```
# store 1 dept 1
training_1.1 <- window(store_1.1_clean, end = c(2012, 26))
validation_1.1 <- window(store_1.1_clean, start = c(2012, 27))

predictors_1.1 <- as.matrix(store_1.1["Temperature"][1:130,])

# store 1 dept 2
training_1.2 <- window(store_1.2_clean, end = c(2012, 26))
```

```
validation_1.2 <- window(store_1.2_clean, start = c(2012, 27))

predictors_1.2 <- as.matrix(store_1.2["Temperature"][1:130,])

# store 1 dept 3
training_1.3 <- window(store_1.3_clean, end = c(2012, 26))
validation_1.3 <- window(store_1.3_clean, start = c(2012, 27))

predictors_1.3 <- as.matrix(store_1.3["Temperature"][1:130,])

# store 1 dept 4
training_1.4 <- window(store_1.4_clean, end = c(2012, 26))
validation_1.4 <- window(store_1.4_clean, start = c(2012, 27))

predictors_1.4 <- as.matrix(store_1.4["Temperature"][1:130,])

 # store 1 dept 5
training_1.5 <- window(store_1.5_clean, end = c(2012, 26))
validation_1.5 <- window(store_1.5_clean, start = c(2012, 27))

predictors_1.5 <- as.matrix(store_1.5["Temperature"][1:130,])

# store 1 dept 6
training_1.6 <- window(store_1.6_clean, end = c(2012, 26))
validation_1.6 <- window(store_1.6_clean, start = c(2012, 27))

predictors_1.6 <- as.matrix(store_1.6["Temperature"][1:130,])

# store 1 dept 7
training_1.7 <- window(store_1.7_clean, end = c(2012, 26))
validation_1.7 <- window(store_1.7_clean, start = c(2012, 27))

predictors_1.7 <- as.matrix(store_1.7["Temperature"][1:130,])

# store 1 dept 8
training_1.8 <- window(store_1.8_clean, end = c(2012, 26))
validation_1.8 <- window(store_1.8_clean, start = c(2012, 27))

predictors_1.8 <- as.matrix(store_1.8["Temperature"][1:130,])

# store 1 dept 9
training_1.9 <- window(store_1.9_clean, end = c(2012, 26))
validation_1.9 <- window(store_1.9_clean, start = c(2012, 27))

predictors_1.9 <- as.matrix(store_1.9["Temperature"][1:130,])

# store 1 dept 10
training_1.10 <- window(store_1.10_clean, end = c(2012, 26))
```

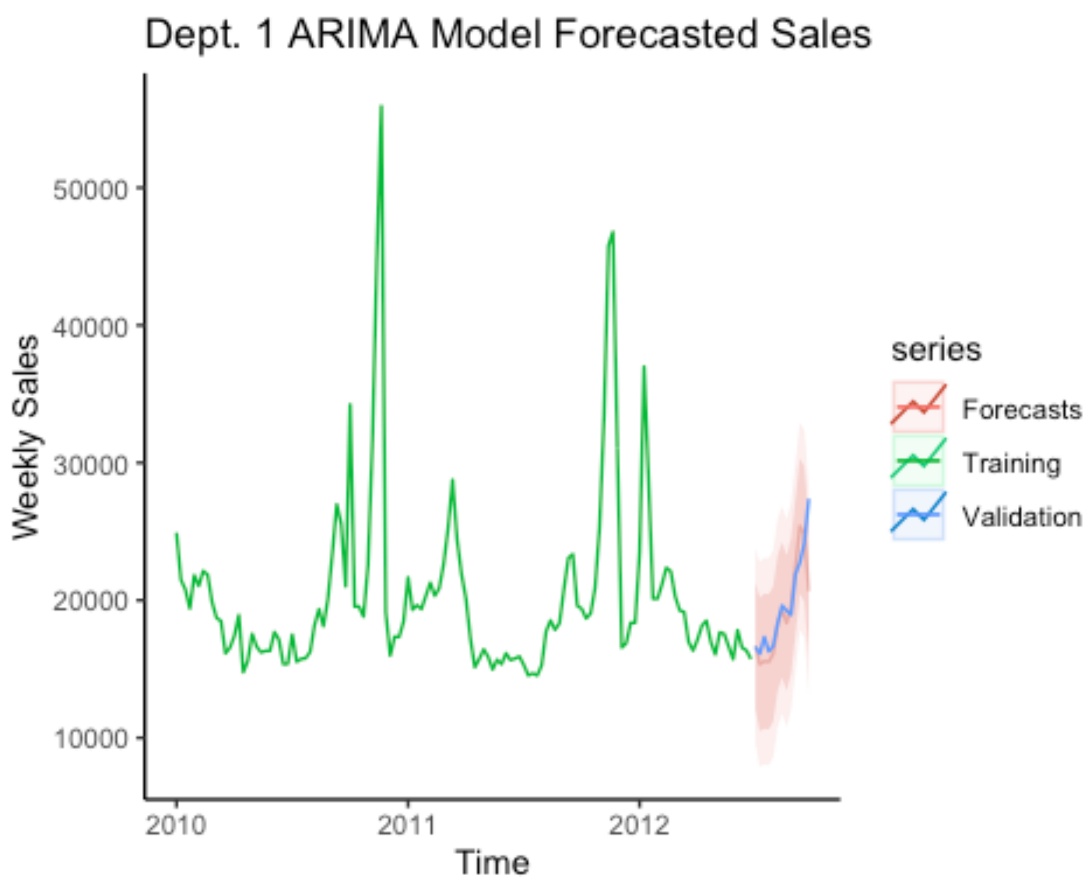
```
validation_1.10 <- window(store_1.10_clean, start = c(2012, 27))

predictors_1.10 <- as.matrix(store_1.10["Temperature"][1:130,])

# store 1 dept 11
training_1.11 <- window(store_1.11_clean, end = c(2012, 26))
validation_1.11 <- window(store_1.11_clean, start = c(2012, 27))

predictors_1.11 <- as.matrix(store_1.11["Temperature"][1:130,])

# store 1 dept 12
training_1.12 <- window(store_1.12_clean, end = c(2012, 26))
validation_1.12 <- window(store_1.12_clean, start = c(2012, 27))

predictors_1.12 <- as.matrix(store_1.12["Temperature"][1:130,])

# store 1 dept 13
training_1.13 <- window(store_1.13_clean, end = c(2012, 26))
validation_1.13 <- window(store_1.13_clean, start = c(2012, 27))

predictors_1.13 <- as.matrix(store_1.13["Temperature"][1:130,])
```

Department 1 Models:

```
AutoArima_1.1 <- auto.arima(training_1.1, xreg = predictors_1.1)
summary(AutoArima_1.1)

## Series: training_1.1
## Regression with ARIMA(1,0,0)(0,1,0)[52] errors
##
## Coefficients:
##          ar1     xreg
##       0.4198   7.6841
## s.e.  0.1023  70.5586
##
## sigma^2 = 15852734:  log likelihood = -756.34
## AIC=1518.67   AICc=1519   BIC=1525.74
##
## Training set error measures:
##                       ME     RMSE      MAE        MPE     MAPE      MASE
## Training set 27.52535 3044.298 1340.801 -0.5772076 6.070496 0.5054659
##                     ACF1
## Training set -0.0297231

# ARIMA model parameters decided by the ACF and PACF plots above
arima_1.1 <- Arima(training_1.1, xreg = predictors_1.1, order = c(0, 1, 3), s
easonal = c(0, 1, 1))
summary(arima_1.1)

## Series: training_1.1
## Regression with ARIMA(0,1,3)(0,1,1)[52] errors
```

```
##
## Coefficients:
##          ma1      ma2      ma3     sma1     xreg
##       -0.682  -0.1859  -0.1321  -0.9992  -0.1983
## s.e.   0.126   0.1570   0.1110   0.6437  76.6634
##
## sigma^2 = 8402521:  log likelihood = -745.56
## AIC=1503.12   AICc=1504.32   BIC=1517.19
##
## Training set error measures:
##                    ME     RMSE      MAE        MPE     MAPE      MASE
## Training set 21.15138 2157.245 954.3406 -0.5351687 4.337541 0.3597749
##                    ACF1
## Training set 0.01322383
```

```
# prediction on the arima
new.predictors_1.1 <- as.matrix(store_1.1["Temperature"][131:143,])
forecast.arima.sales_1.1 <- forecast(arima_1.1, xreg = new.predictors_1.1)

# plot of forecasted values
autoplot(training_1.1, series = "Training") +
  autolayer(forecast.arima.sales_1.1, alpha = 0.3, series = "Forecasts") +
  autolayer(validation_1.1, series = "Validation") +
  labs(title = "Dept. 1 ARIMA Model Forecasted Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```




```
# linear model
temp_1.1 <- store_1.1[1:130, 6]
linear_1.1 <- tslm(training_1.1 ~ trend + season + temp_1.1)
summary(linear_1.1)

##
## Call:
## tslm(formula = training_1.1 ~ trend + season + temp_1.1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7472.5  -923.2  -149.6   743.6 10600.4
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept) 21569.828   3822.259   5.643 2.73e-07 ***
## trend           4.177      7.414   0.563  0.57486
## season2      2767.099   2484.286   1.114  0.26886
## season3      -298.131   2442.879  -0.122  0.90319
## season4     -4035.299   2522.459  -1.600  0.11380
## season5     -2916.729   2523.637  -1.156  0.25140
## season6     -2556.742   2543.324  -1.005  0.31795
## season7     -2236.600   2640.554  -0.847  0.39964
```

```
## season8     -2297.040   2692.077  -0.853  0.39620
## season9     -3009.648   2709.301  -1.111  0.27013
## season10    -3025.775   2889.464  -1.047  0.29834
## season11    -2004.092   2922.450  -0.686  0.49495
## season12    -4932.793   2893.754  -1.705  0.09235 .
## season13    -5876.320   2920.762  -2.012  0.04777 *
## season14    -5987.215   3004.315  -1.993  0.04987 *
## season15    -6244.059   3176.984  -1.965  0.05302 .
## season16    -8105.091   3026.936  -2.678  0.00908 **
## season17    -8390.852   3358.453  -2.498  0.01463 *
## season18    -7874.205   3465.387  -2.272  0.02590 *
## season19    -7837.702   3468.078  -2.260  0.02669 *
## season20    -8409.296   3610.211  -2.329  0.02250 *
## season21    -8409.806   3544.694  -2.373  0.02020 *
## season22    -8878.636   3634.651  -2.443  0.01690 *
## season23    -7430.336   3570.596  -2.081  0.04080 *
## season24    -8207.939   3586.248  -2.289  0.02487 *
## season25    -8852.632   3598.520  -2.460  0.01616 *
## season26    -9023.249   3629.807  -2.486  0.01512 *
## season27    -8383.836   4167.171  -2.012  0.04778 *
## season28    -9734.313   4136.279  -2.353  0.02119 *
## season29    -9542.306   4102.249  -2.326  0.02268 *
## season30    -9538.439   4007.161  -2.380  0.01980 *
## season31    -8879.376   3894.627  -2.280  0.02542 *
## season32    -6432.042   3533.167  -1.820  0.07262 .
## season33    -5585.336   3711.366  -1.505  0.13649
## season34    -6454.529   3579.533  -1.803  0.07532 .
## season35    -5147.264   3457.600  -1.489  0.14071
## season36    -1967.422   3086.353  -0.637  0.52574
## season37      868.415   3189.152   0.272  0.78613
## season38      373.312   3090.523   0.121  0.90417
## season39    -3808.036   3136.767  -1.214  0.22851
## season40     3061.973   2821.545   1.085  0.28126
## season41    -4718.474   2873.096  -1.642  0.10466
## season42    -4455.624   2820.498  -1.580  0.11832
## season43    -4080.003   2947.711  -1.384  0.17037
## season44      399.048   2736.163   0.146  0.88443
## season45     9023.771   2739.765   3.294  0.00150 **
## season46    21773.015   2744.464   7.933 1.48e-11 ***
## season47    27805.645   2741.263  10.143 8.89e-16 ***
## season48     1666.743   2736.311   0.609  0.54426
## season49    -7236.324   2736.449  -2.644  0.00994 **
## season50    -6165.837   2768.465  -2.227  0.02890 *
## season51    -5681.928   2738.220  -2.075  0.04137 *
## season52    -5118.715   2738.519  -1.869  0.06545 .
## temp_1.1       33.495     74.303   0.451  0.65343
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2992 on 76 degrees of freedom
```

```
## Multiple R-squared:  0.8763, Adjusted R-squared:   0.79
## F-statistic: 10.16 on 53 and 76 DF,  p-value: < 2.2e-16
```

```
# calculating RMSE
sqrt(mean(linear_1.1$residuals^2))
```

```
## [1] 2287.37
```

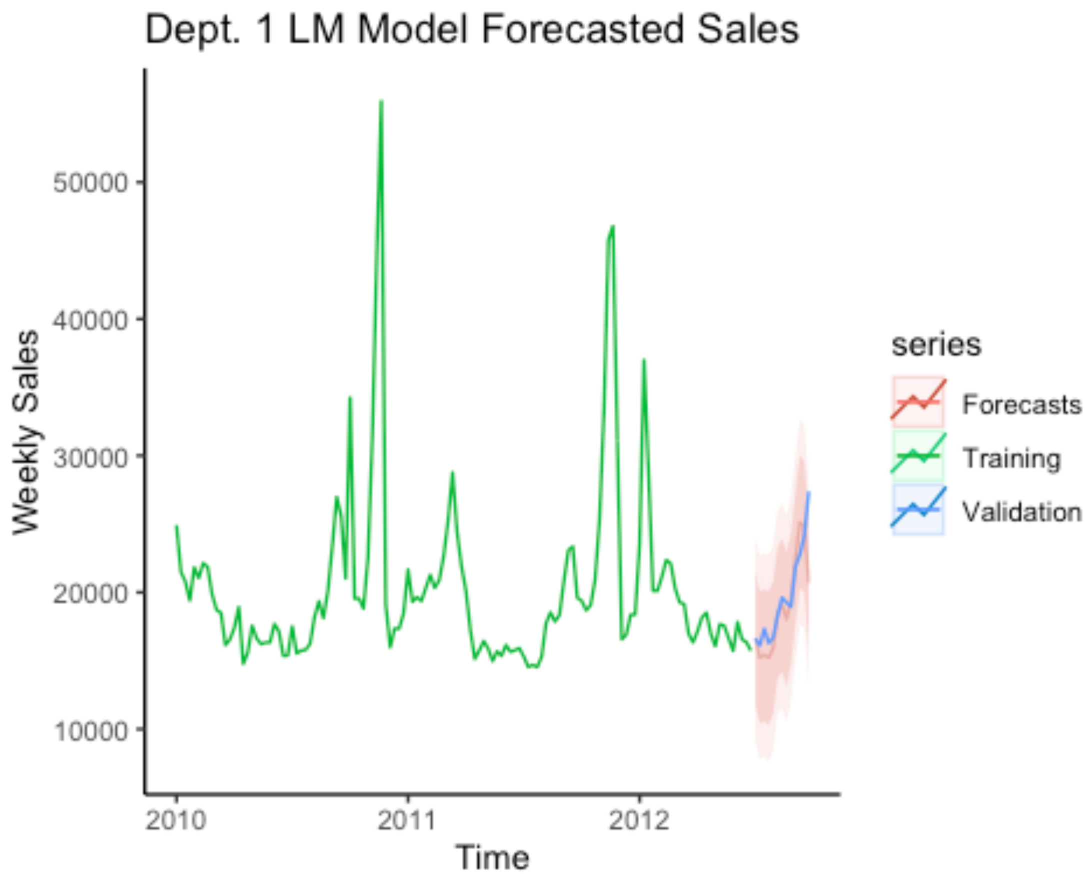
```
# forecasting
temp.new_1.1 <- store_1.1[131:143, 6]
forecast.lm.sales_1.1 <- forecast(linear_1.1, temp.new_1.1, h = 13)
```

```
## Warning in forecast.lm(linear_1.1, temp.new_1.1, h = 13): newdata column n
ames
## not specified, defaulting to first variable required.
```

```
forecast.lm.sales_1.1
```

```
##          Point Forecast    Lo 80    Hi 80     Lo 95    Hi 95
## 2012.500       16617.41 11798.29 21436.54  9193.071 24041.76
## 2012.519       15235.61 10410.67 20060.54  7802.313 22668.90
## 2012.538       15425.09 10604.40 20245.79  7998.326 22851.86
## 2012.558       15192.31 10285.23 20099.39  7632.464 22752.16
## 2012.577       15950.34 11123.09 20777.59  8513.484 23387.20
## 2012.596       18518.08 13707.58 23328.58 11107.018 25929.14
## 2012.615       19067.84 14212.97 23922.71 11588.433 26547.25
## 2012.635       18032.00 13133.69 22930.32 10485.658 25578.35
## 2012.654       19551.45 14757.21 24345.68 12165.448 26937.45
## 2012.673       22483.25 17692.18 27274.32 15102.133 29864.37
## 2012.692       25137.03 20275.56 29998.50 17647.451 32626.61
## 2012.712       24812.91 20021.05 29604.78 17430.565 32195.26
## 2012.731       20675.60 15883.49 25467.71 13292.870 28058.33
```

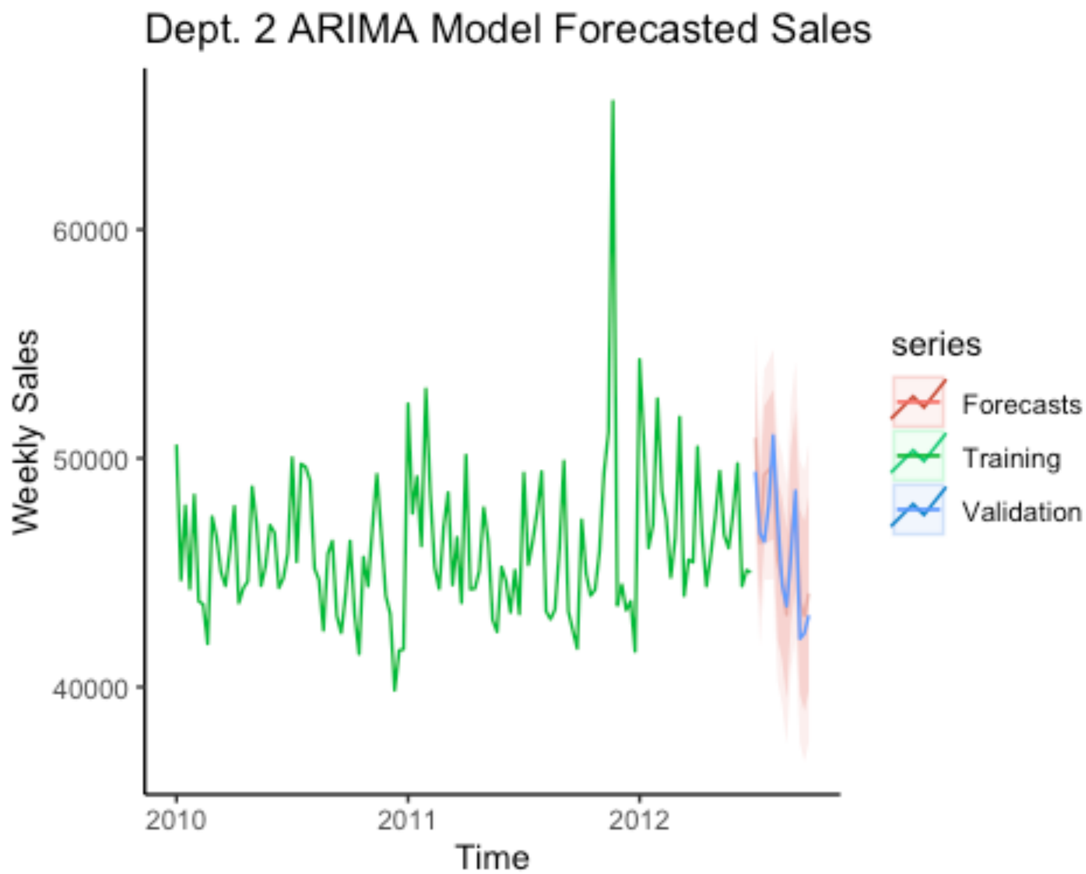
```
# plot of forecasted values
autoplot(training_1.1, series = "Training") +
  autolayer(forecast.lm.sales_1.1, alpha = 0.3, series = "Forecasts") +
  autolayer(validation_1.1, series = "Validation") +
  labs(title = "Dept. 1 LM Model Forecasted Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```



Department 2 Models:

```
# Auto ARIMA model
AutoArima_1.2 <- auto.arima(training_1.2, xreg = predictors_1.2)
summary(AutoArima_1.2)

## Series: training_1.2
## Regression with ARIMA(1,0,1)(0,1,0)[52] errors
##
## Coefficients:
##          ar1      ma1     xreg
##       0.9108  -0.7351  17.0723
## s.e.  0.0798   0.1174  48.1717
##
## sigma^2 = 7731094:  log likelihood = -727.88
## AIC=1463.75   AICc=1464.3   BIC=1473.18
##
## Training set error measures:
##                    ME     RMSE      MAE       MPE     MAPE      MASE
ACF1
## Training set 224.2849 2111.93 1020.085 0.3279628 2.111287 0.5438332 -0.099
5141
```

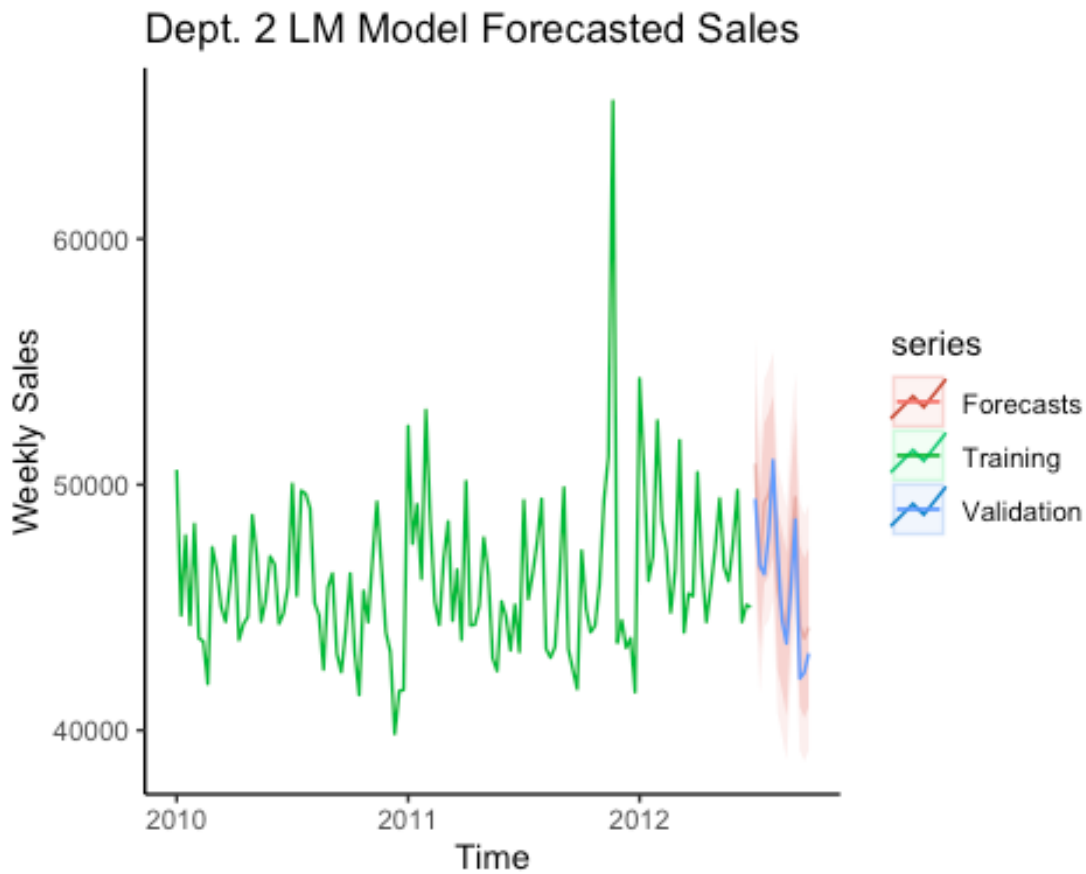
```r
# ARIMA model parameters decided by the ACF and PACF plots above
arima_1.2 <- Arima(training_1.2, xreg = predictors_1.2, order = c(4, 1, 0), s
easonal = c(1, 1, 1))
summary(arima_1.2)

## Series: training_1.2
## Regression with ARIMA(4,1,0)(1,1,1)[52] errors
##
## Coefficients:
##           ar1      ar2      ar3      ar4      sar1      sma1     xreg
##       -0.8705  -0.6119  -0.3698  -0.1716   -0.7336    0.0423  25.5466
## s.e.   0.1159   0.1542   0.1661   0.1183  378.7502  771.4222  59.4157
##
## sigma^2 = 5021841:  log likelihood = -718.54
## AIC=1453.08   AICc=1455.2   BIC=1471.83
##
## Training set error measures:
##                     ME     RMSE      MAE        MPE     MAPE      MASE
## Training set -61.27756 1644.405 851.6671 -0.2511672 1.787644 0.4540452
##                     ACF1
## Training set -0.04005042

# prediction on the arima
new.predictors_1.2 <- as.matrix(store_1.2["Temperature"][131:143,])
forecast.arima.sales_1.2 <- forecast(arima_1.2, xreg = new.predictors_1.2)
forecast.arima.sales_1.2

##          Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## 2012.500       50894.45 48021.25 53767.65 46500.27 55288.63
## 2012.519       46311.92 43414.75 49209.09 41881.08 50742.76
## 2012.538       49283.15 46279.84 52286.45 44689.99 53876.31
## 2012.558       49483.43 46349.86 52617.00 44691.05 54275.81
## 2012.577       49735.91 46452.64 53019.19 44714.58 54757.25
## 2012.596       45569.36 42101.47 49037.24 40265.69 50873.03
## 2012.615       44724.58 41157.54 48291.61 39269.27 50179.88
## 2012.635       43170.47 39479.83 46861.12 37526.12 48814.82
## 2012.654       46638.69 42826.09 50451.28 40807.82 52469.55
## 2012.673       48172.38 44240.66 52104.10 42159.33 54185.43
## 2012.692       43732.72 39685.93 47779.51 37543.69 49921.74
## 2012.712       43098.29 38944.87 47251.71 36746.18 49450.39
## 2012.731       44099.77 39838.30 48361.24 37582.42 50617.13

# plot of forecasted values
autoplot(training_1.2, series = "Training") +
  autolayer(forecast.arima.sales_1.2, alpha = 0.3, series = "Forecasts") +
  autolayer(validation_1.2, series = "Validation") +
  labs(title = "Dept. 2 ARIMA Model Forecasted Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```




```
# linear model
temp_1.2 <- store_1.2[1:130, 6]
linear_1.2 <- tslm(training_1.2 ~ trend + season + temp_1.2)
summary(linear_1.2)

##
## Call:
## tslm(formula = training_1.2 ~ trend + season + temp_1.2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9111.1  -663.8    79.7   745.8  9111.1
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    49974.56    2587.67  19.313  < 2e-16 ***
## trend             16.33       5.02   3.253 0.001707 **
## season2        -4627.50    1681.86  -2.751 0.007415 **
## season3        -4747.56    1653.83  -2.871 0.005304 **
## season4        -6930.13    1707.70  -4.058 0.000119 ***
## season5        -1448.14    1708.50  -0.848 0.399319
## season6        -5919.43    1721.83  -3.438 0.000954 ***
## season7        -7603.63    1787.65  -4.253 5.93e-05 ***
```

```
## season8      -9413.59    1822.54  -5.165 1.87e-06 ***
## season9      -6119.28    1834.19  -3.336 0.001316 **
## season10     -4344.38    1956.16  -2.221 0.029339 *
## season11     -8848.15    1978.50  -4.472 2.67e-05 ***
## season12     -7826.53    1959.07  -3.995 0.000148 ***
## season13     -8337.92    1977.35  -4.217 6.77e-05 ***
## season14     -3944.50    2033.92  -1.939 0.056169 .
## season15     -8716.31    2150.82  -4.053 0.000121 ***
## season16     -9166.86    2049.23  -4.473 2.66e-05 ***
## season17     -8595.20    2273.67  -3.780 0.000310 ***
## season18     -5843.93    2346.07  -2.491 0.014922 *
## season19     -6239.74    2347.89  -2.658 0.009589 **
## season20     -9309.12    2444.11  -3.809 0.000281 ***
## season21     -9356.86    2399.76  -3.899 0.000207 ***
## season22     -7375.48    2460.66  -2.997 0.003678 **
## season23     -6955.10    2417.29  -2.877 0.005205 **
## season24    -10027.00    2427.89  -4.130 9.23e-05 ***
## season25     -9060.70    2436.20  -3.719 0.000380 ***
## season26     -9414.54    2457.38  -3.831 0.000261 ***
## season27     -4179.39    2821.17  -1.481 0.142625
## season28     -8464.83    2800.26  -3.023 0.003412 **
## season29     -5797.45    2777.22  -2.088 0.040194 *
## season30     -5186.81    2712.84  -1.912 0.059653 .
## season31     -4557.19    2636.66  -1.728 0.087979 .
## season32     -9298.15    2391.95  -3.887 0.000215 ***
## season33     -9865.79    2512.59  -3.927 0.000188 ***
## season34    -10680.56    2423.34  -4.407 3.39e-05 ***
## season35     -7606.46    2340.79  -3.250 0.001723 **
## season36     -5114.41    2089.46  -2.448 0.016684 *
## season37    -10152.68    2159.05  -4.702 1.13e-05 ***
## season38    -10857.20    2092.28  -5.189 1.70e-06 ***
## season39    -10466.05    2123.59  -4.928 4.73e-06 ***
## season40     -6134.95    1910.18  -3.212 0.001935 **
## season41     -9018.20    1945.08  -4.636 1.45e-05 ***
## season42    -10293.60    1909.48  -5.391 7.62e-07 ***
## season43     -8246.38    1995.60  -4.132 9.15e-05 ***
## season44     -7617.41    1852.38  -4.112 9.82e-05 ***
## season45     -4509.26    1854.82  -2.431 0.017409 *
## season46     -2652.14    1858.00  -1.427 0.157555
## season47      3285.18    1855.83   1.770 0.080705 .
## season48     -8996.40    1852.48  -4.856 6.25e-06 ***
## season49     -9008.51    1852.58  -4.863 6.10e-06 ***
## season50    -11034.82    1874.25  -5.888 9.99e-08 ***
## season51    -10222.56    1853.77  -5.514 4.62e-07 ***
## season52    -11309.46    1853.98  -6.100 4.11e-08 ***
## temp_1.2        33.95      50.30   0.675 0.501785
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2025 on 76 degrees of freedom
```

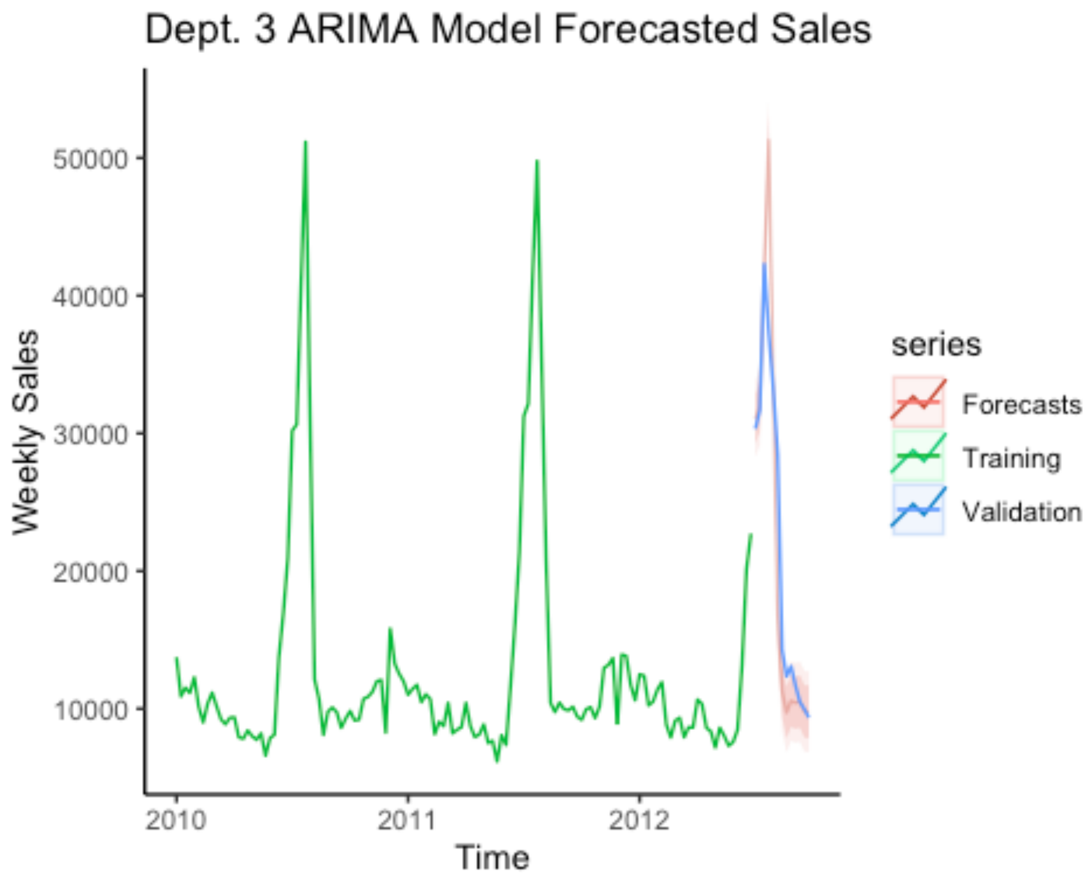
```
## Multiple R-squared:  0.7667, Adjusted R-squared:  0.6041
## F-statistic: 4.713 on 53 and 76 DF,  p-value: 5.202e-10

# calculating RMSE
sqrt(mean(linear_1.2$residuals^2))

## [1] 1548.548

# forecasting
temp.new_1.2 <- store_1.2[131:143, 6]
forecast.lm.sales_1.2 <- forecast(linear_1.2, temp.new_1.2, h = 13)

## Warning in forecast.lm(linear_1.2, temp.new_1.2, h = 13): newdata column n
ames
## not specified, defaulting to first variable required.

forecast.lm.sales_1.2

##          Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## 2012.500       50857.34 47594.79 54119.88 45831.06 55883.61
## 2012.519       46552.24 43285.76 49818.72 41519.90 51584.57
## 2012.538       49229.15 45965.54 52492.76 44201.23 54257.07
## 2012.558       49612.02 46289.93 52934.11 44494.01 54730.03
## 2012.577       50354.04 47086.00 53622.09 45319.30 55388.79
## 2012.596       45747.21 42490.50 49003.92 40729.93 50764.50
## 2012.615       44890.69 41603.95 48177.44 39827.14 49954.25
## 2012.635       43919.10 40602.95 47235.26 38810.23 49027.97
## 2012.654       47220.36 43974.66 50466.05 42220.04 52220.68
## 2012.673       49473.10 46229.55 52716.65 44476.09 54470.11
## 2012.692       44262.39 40971.18 47553.60 39191.95 49332.83
## 2012.712       43743.26 40499.18 46987.35 38745.42 48741.11
## 2012.731       44191.14 40946.88 47435.40 39193.04 49189.24

# plot of forecasted values
autoplot(training_1.2, series = "Training") +
  autolayer(forecast.lm.sales_1.2, alpha = 0.3, series = "Forecasts") +
  autolayer(validation_1.2, series = "Validation") +
  labs(title = "Dept. 2 LM Model Forecasted Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```



Department 3 Models:

```
# Auto ARIMA model
AutoArima_1.3 <- auto.arima(training_1.3, xreg = predictors_1.3)
summary(AutoArima_1.3)

## Series: training_1.3
## Regression with ARIMA(0,1,2)(0,1,0)[52] errors
##
## Coefficients:
##           ma1      ma2      xreg
##       -0.6785  -0.2514  -12.5049
## s.e.   0.1276   0.1345   29.4822
##
## sigma^2 = 2404864:  log likelihood = -674.33
## AIC=1356.66   AICc=1357.22   BIC=1366.04
##
## Training set error measures:
##                   ME     RMSE       MAE       MPE     MAPE      MASE
ACF1
## Training set 149.289 1170.01 612.0587 0.4733439 5.184392 0.5780193 -0.0269
8826
```

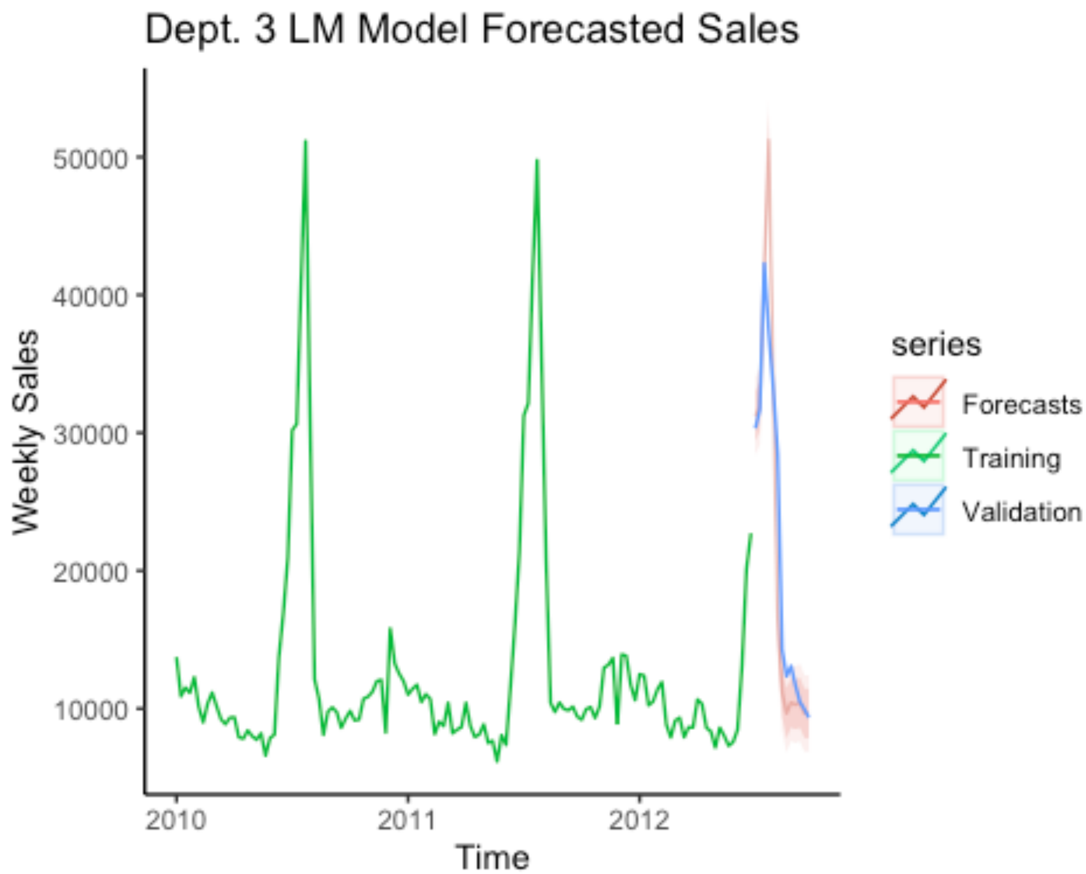
```
# ARIMA model parameters decided by the ACF and PACF plots above
arima_1.3 <- Arima(training_1.3, xreg = predictors_1.3, order = c(0, 1, 3), s
easonal = c(0, 1, 1))
summary(arima_1.3)

## Series: training_1.3
## Regression with ARIMA(0,1,3)(0,1,1)[52] errors
##
## Coefficients:
##           ma1      ma2     ma3     sma1      xreg
##       -0.7139  -0.3030  0.1025  -0.9968  -38.8556
## s.e.   0.1171   0.1334  0.1131   0.8625   31.8638
##
## sigma^2 = 1284735:  log likelihood = -672.07
## AIC=1356.15   AICc=1357.35   BIC=1370.21
##
## Training set error measures:
##                    ME     RMSE      MAE        MPE    MAPE      MASE
ACF1
## Training set 88.35297 843.5314 458.2682 0.09653082 3.90974 0.4327819 0.010
85711

# prediction on the arima
new.predictors_1.3 <- as.matrix(store_1.3["Temperature"][131:143,])
forecast.arima.sales_1.3 <- forecast(arima_1.3, xreg = new.predictors_1.3)
forecast.arima.sales_1.3

##          Point Forecast     Lo 80    Hi 80     Lo 95    Hi 95
## 2012.500      31034.176 29257.494 32810.86 28316.976 33751.38
## 2012.519      31974.470 30126.314 33822.63 29147.960 34800.98
## 2012.538      41954.948 40106.676 43803.22 39128.259 44781.64
## 2012.558      51282.888 49428.374 53137.40 48446.654 54119.12
## 2012.577      33677.978 31817.244 35538.71 30832.230 36523.73
## 2012.596      16686.518 14819.584 18553.45 13831.288 19541.75
## 2012.615      11271.535  9398.422 13144.65  8406.856 14136.22
## 2012.635       9725.615  7846.343 11604.89  6851.516 12599.71
## 2012.654      10562.334  8676.923 12447.75  7678.847 13445.82
## 2012.673      10423.491  8531.961 12315.02  7530.646 13316.34
## 2012.692      10522.976  8625.347 12420.60  7620.803 13425.15
## 2012.712       9798.161  7894.453 11701.87  6886.691 12709.63
## 2012.731       9779.762  7869.993 11689.53  6859.023 12700.50

# plot of forecasted values
autoplot(training_1.3, series = "Training") +
  autolayer(forecast.arima.sales_1.3, alpha = 0.3, series = "Forecasts") +
  autolayer(validation_1.3, series = "Validation") +
  labs(title = "Dept. 3 ARIMA Model Forecasted Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```




```
# linear model
temp_1.3 <- store_1.3[1:130, 6]
linear_1.3 <- tslm(training_1.3 ~ trend + season + temp_1.3)
summary(linear_1.3)

##
## Call:
## tslm(formula = training_1.3 ~ trend + season + temp_1.3)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4164.1  -472.2    -9.6   460.7  4164.1
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept) 14388.083   1429.794  10.063 1.26e-15 ***
## trend           4.349      2.774   1.568  0.12103
## season2     -1133.453    929.298  -1.220  0.22636
## season3     -1236.296    913.809  -1.353  0.18010
## season4     -1321.177    943.577  -1.400  0.16553
## season5      -457.909    944.018  -0.485  0.62903
## season6     -1068.569    951.382  -1.123  0.26490
## season7     -3133.615    987.753  -3.172  0.00218 **
```

```
## season8     -2612.164   1007.027  -2.594  0.01138 *
## season9     -2025.140   1013.469  -1.998  0.04927 *
## season10    -1490.011   1080.863  -1.379  0.17208
## season11    -2976.271   1093.202  -2.723  0.00803 **
## season12    -2807.136   1082.468  -2.593  0.01140 *
## season13    -2565.086   1092.571  -2.348  0.02149 *
## season14    -1200.904   1123.825  -1.069  0.28864
## season15    -2177.171   1188.416  -1.832  0.07087 .
## season16    -3211.965   1132.287  -2.837  0.00584 **
## season17    -2716.416   1256.298  -2.162  0.03375 *
## season18    -2905.476   1296.299  -2.241  0.02792 *
## season19    -2961.376   1297.305  -2.283  0.02524 *
## season20    -2837.211   1350.473  -2.101  0.03897 *
## season21    -4169.770   1325.965  -3.145  0.00237 **
## season22    -2946.253   1359.615  -2.167  0.03337 *
## season23    -2855.757   1335.654  -2.138  0.03572 *
## season24     1988.656   1341.510   1.482  0.14237
## season25     6886.377   1346.100   5.116 2.28e-06 ***
## season26    10855.544   1357.804   7.995 1.13e-11 ***
## season27    20292.262   1558.815  13.018  < 2e-16 ***
## season28    20939.220   1547.260  13.533  < 2e-16 ***
## season29    30865.227   1534.530  20.114  < 2e-16 ***
## season30    39895.758   1498.960  26.616  < 2e-16 ***
## season31    22379.807   1456.865  15.362  < 2e-16 ***
## season32     5461.171   1321.653   4.132 9.16e-05 ***
## season33     -277.896   1388.312  -0.200  0.84188
## season34    -2047.662   1338.997  -1.529  0.13035
## season35     -994.723   1293.386  -0.769  0.44423
## season36    -1504.155   1154.514  -1.303  0.19656
## season37    -1602.247   1192.968  -1.343  0.18324
## season38    -2159.402   1156.073  -1.868  0.06563 .
## season39    -2125.308   1173.372  -1.811  0.07405 .
## season40    -2501.849   1055.457  -2.370  0.02031 *
## season41    -2342.982   1074.740  -2.180  0.03235 *
## season42    -2365.928   1055.065  -2.242  0.02785 *
## season43    -1753.443   1102.652  -1.590  0.11594
## season44    -1889.080   1023.518  -1.846  0.06883 .
## season45     -493.135   1024.865  -0.481  0.63178
## season46      261.103   1026.623   0.254  0.79993
## season47      519.765   1025.426   0.507  0.61371
## season48    -3972.790   1023.573  -3.881  0.00022 ***
## season49     2433.476   1023.625   2.377  0.01996 *
## season50      798.234   1035.601   0.771  0.44322
## season51     -304.411   1024.287  -0.297  0.76713
## season52    -1129.421   1024.399  -1.103  0.27371
## temp_1.3      -46.888     27.794  -1.687  0.09571 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1119 on 76 degrees of freedom
```

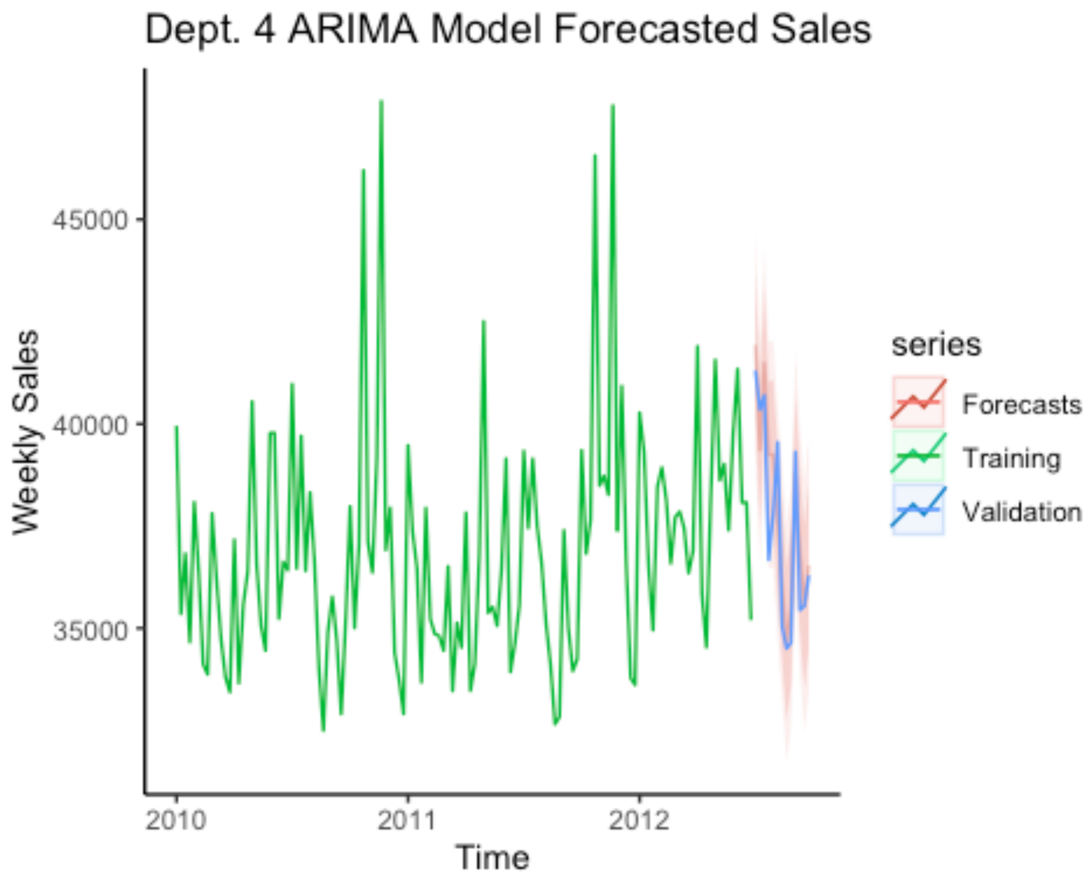
```
## Multiple R-squared:  0.9886, Adjusted R-squared:  0.9806
## F-statistic:   124 on 53 and 76 DF,  p-value: < 2.2e-16

# calculating RMSE
sqrt(mean(linear_1.3$residuals^2))

## [1] 855.6376

# forecasting
temp.new_1.3 <- store_1.3[131:143, 6]
forecast.lm.sales_1.3 <- forecast(linear_1.3, temp.new_1.3, h = 13)

## Warning in forecast.lm(linear_1.3, temp.new_1.3, h = 13): newdata column n
ames
## not specified, defaulting to first variable required.

forecast.lm.sales_1.3

##          Point Forecast     Lo 80    Hi 80     Lo 95    Hi 95
## 2012.500      31212.527 29409.835 33015.22 28435.300 33989.75
## 2012.519      31913.535 30108.670 33718.40 29132.959 34694.11
## 2012.538      41853.268 40049.987 43656.55 39075.134 44631.40
## 2012.558      51225.275 49389.681 53060.87 48397.360 54053.19
## 2012.577      33580.979 31775.248 35386.71 30799.071 36362.89
## 2012.596      16503.990 14704.523 18303.46 13731.731 19276.25
## 2012.615      11190.796  9374.734 13006.86  8392.970 13988.62
## 2012.635       9664.508  7832.193 11496.82  6841.643 12487.37
## 2012.654      10430.621  8637.239 12224.00  7667.737 13193.51
## 2012.673      10278.605  8486.409 12070.80  7517.548 13039.66
## 2012.692      10445.560  8627.027 12264.09  7643.929 13247.19
## 2012.712       9659.252  7866.757 11451.75  6897.734 12420.77
## 2012.731       9641.897  7849.309 11434.49  6880.236 12403.56

# plot of forecasted values
autoplot(training_1.3, series = "Training") +
  autolayer(forecast.lm.sales_1.3, alpha = 0.3, series = "Forecasts") +
  autolayer(validation_1.3, series = "Validation") +
  labs(title = "Dept. 3 LM Model Forecasted Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```



Department 4 Models:

```
# Auto ARIMA model
AutoArima_1.4 <- auto.arima(training_1.4, xreg = predictors_1.4)
summary(AutoArima_1.4)

## Series: training_1.4
## Regression with ARIMA(0,1,1)(0,1,0)[52] errors
##
## Coefficients:
##           ma1     xreg
##       -0.8408  13.6148
## s.e.   0.0543  25.3298
##
## sigma^2 = 2104781:  log likelihood = -669.39
## AIC=1344.78   AICc=1345.11   BIC=1351.81
##
## Training set error measures:
##                    ME     RMSE      MAE       MPE     MAPE      MASE
ACF1
## Training set 110.4931 1101.951 700.5318 0.2553229 1.897304 0.4728834 -0.07
66048
```

```r
# ARIMA model parameters decided by the ACF and PACF plots above
arima_1.4 <- Arima(training_1.4, xreg = predictors_1.4, order = c(1, 1, 3), s
easonal = c(0, 1, 1))
summary(arima_1.4)

## Series: training_1.4
## Regression with ARIMA(1,1,3)(0,1,1)[52] errors
##
## Coefficients:
##          ar1      ma1     ma2     ma3     sma1     xreg
##       0.6045  -1.4955  0.4865  0.0802  -0.4037  -6.5806
## s.e.  1.1541   1.1686  1.1787  0.2237   0.3027  30.8903
##
## sigma^2 = 1890203:  log likelihood = -667.46
## AIC=1348.93   AICc=1350.55   BIC=1365.33
##
## Training set error measures:
##                     ME     RMSE      MAE       MPE    MAPE      MASE
ACF1
## Training set 76.57734 1016.042 641.7766 0.1695963 1.73829 0.4332216 -0.015
4414

# prediction on the arima
new.predictors_1.4 <- as.matrix(store_1.4["Temperature"][131:143,])
forecast.arima.sales_1.4 <- forecast(arima_1.4, xreg = new.predictors_1.4)

# plot of forecasted values
autoplot(training_1.4, series = "Training") +
  autolayer(forecast.arima.sales_1.4, alpha = 0.3, series = "Forecasts") +
  autolayer(validation_1.4, series = "Validation") +
  labs(title = "Dept. 4 ARIMA Model Forecasted Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```




```
# linear model
temp_1.4 <- store_1.4[1:130, 6]
linear_1.4 <- tslm(training_1.4 ~ trend + season + temp_1.4)
summary(linear_1.4)

##
## Call:
## tslm(formula = training_1.4 ~ trend + season + temp_1.4)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2326.1  -601.8     0.0   551.0  2237.4
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 40489.861   1495.434  27.076  < 2e-16 ***
## trend          16.119      2.901   5.557 3.89e-07 ***
## season2     -2760.029    971.961  -2.840 0.005792 **
## season3     -3299.123    955.760  -3.452 0.000913 ***
## season4     -5256.556    986.895  -5.326 9.87e-07 ***
## season5     -1537.037    987.356  -1.557 0.123692
## season6     -2844.817    995.059  -2.859 0.005483 **
## season7     -3876.691   1033.100  -3.752 0.000340 ***
```

```
## season8     -4450.499   1053.258  -4.225 6.56e-05 ***
## season9     -2879.837   1059.996  -2.717 0.008157 **
## season10    -2554.240   1130.484  -2.259 0.026723 *
## season11    -4173.172   1143.389  -3.650 0.000479 ***
## season12    -4332.760   1132.162  -3.827 0.000264 ***
## season13    -4470.455   1142.729  -3.912 0.000198 ***
## season14     -413.912   1175.418  -0.352 0.725708
## season15    -4953.967   1242.974  -3.986 0.000153 ***
## season16    -4645.005   1184.269  -3.922 0.000191 ***
## season17    -2047.820   1313.973  -1.558 0.123271
## season18     2393.859   1355.810   1.766 0.081474 .
## season19    -2303.328   1356.863  -1.698 0.093686 .
## season20    -2560.992   1412.471  -1.813 0.073760 .
## season21    -3508.654   1386.838  -2.530 0.013477 *
## season22     -450.944   1422.033  -0.317 0.752028
## season23      927.442   1396.972   0.664 0.508767
## season24    -3423.996   1403.096  -2.440 0.017004 *
## season25    -2734.077   1407.897  -1.942 0.055848 .
## season26    -3452.663   1420.138  -2.431 0.017404 *
## season27     1555.223   1630.378   0.954 0.343157
## season28    -1677.688   1618.292  -1.037 0.303162
## season29      759.930   1604.978   0.473 0.637226
## season30    -1786.083   1567.775  -1.139 0.258178
## season31    -1360.134   1523.747  -0.893 0.374876
## season32    -3196.986   1382.328  -2.313 0.023444 *
## season33    -4901.509   1452.048  -3.376 0.001163 **
## season34    -6476.398   1400.469  -4.624 1.51e-05 ***
## season35    -5276.291   1352.763  -3.900 0.000206 ***
## season36    -2860.148   1207.515  -2.369 0.020396 *
## season37    -4527.577   1247.735  -3.629 0.000514 ***
## season38    -6044.145   1209.147  -4.999 3.60e-06 ***
## season39    -4723.322   1227.240  -3.849 0.000246 ***
## season40    -1141.590   1103.911  -1.034 0.304354
## season41    -3831.430   1124.080  -3.409 0.001048 **
## season42    -2474.849   1103.501  -2.243 0.027829 *
## season43     6692.880   1153.273   5.803 1.42e-07 ***
## season44    -2312.054   1070.506  -2.160 0.033943 *
## season45    -2702.945   1071.915  -2.522 0.013776 *
## season46    -1383.474   1073.753  -1.288 0.201499
## season47     7711.445   1072.501   7.190 3.84e-10 ***
## season48    -3113.878   1070.564  -2.909 0.004757 **
## season49     -769.766   1070.618  -0.719 0.474350
## season50    -4939.298   1083.144  -4.560 1.92e-05 ***
## season51    -6467.549   1071.310  -6.037 5.36e-08 ***
## season52    -6982.584   1071.428  -6.517 7.03e-09 ***
## temp_1.4      -30.680     29.070  -1.055 0.294605
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1170 on 76 degrees of freedom
```

```
## Multiple R-squared:  0.9016, Adjusted R-squared:  0.8329
## F-statistic: 13.13 on 53 and 76 DF,  p-value: < 2.2e-16
```

```
# calculating RMSE
sqrt(mean(linear_1.4$residuals^2))
```

```
## [1] 894.9186
```

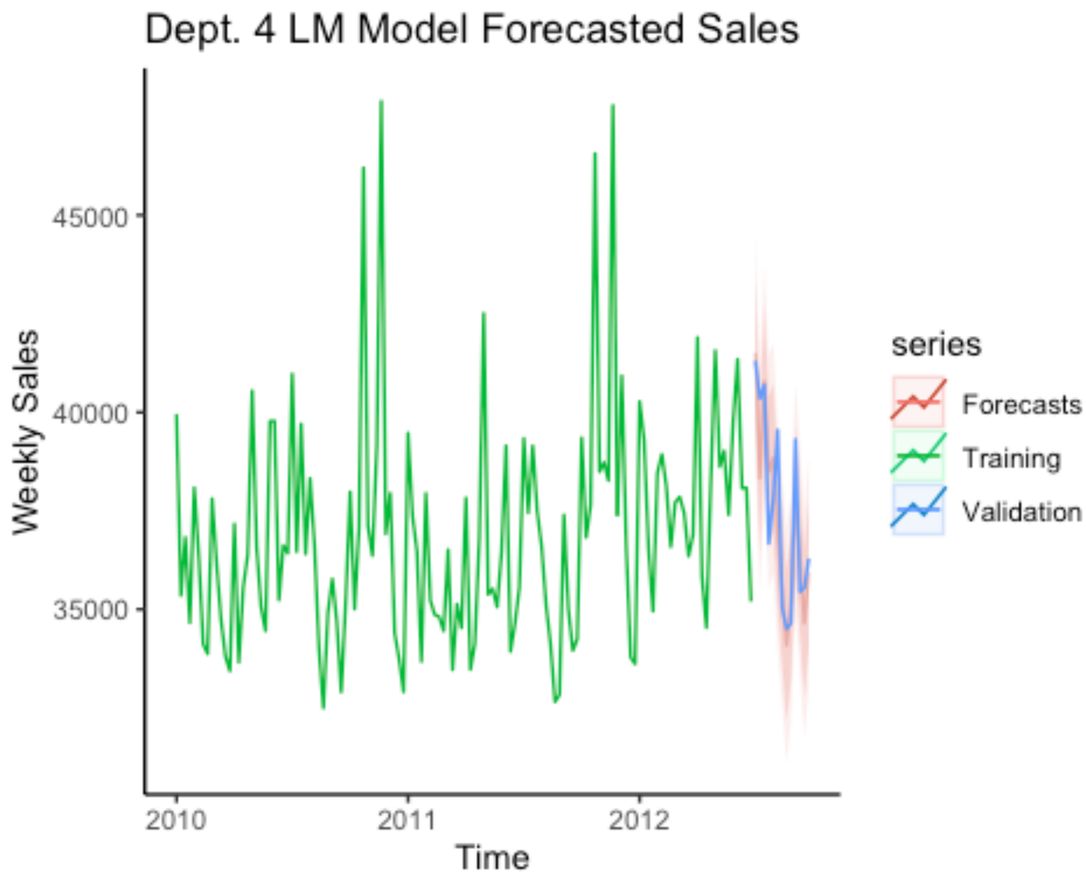
```
# forecasting
temp.new_1.4 <- store_1.4[131:143, 6]
forecast.lm.sales_1.4 <- forecast(linear_1.4, temp.new_1.4, h = 13)
```

```
## Warning in forecast.lm(linear_1.4, temp.new_1.4, h = 13): newdata column n
ames
## not specified, defaulting to first variable required.
```

```
forecast.lm.sales_1.4
```

```
##          Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## 2012.500       41514.88 39629.43 43400.33 38610.16 44419.61
## 2012.519       38330.61 36442.88 40218.33 35422.38 41238.84
## 2012.538       40790.48 38904.42 42676.55 37884.81 43696.16
## 2012.558       38481.18 36561.31 40401.04 35523.44 41438.92
## 2012.577       38836.42 36947.79 40725.05 35926.80 41746.04
## 2012.596       36909.23 35027.15 38791.31 34009.70 39808.76
## 2012.615       35496.64 33597.20 37396.07 32570.37 38422.91
## 2012.635       34094.33 32177.90 36010.77 31141.88 37046.79
## 2012.654       35120.04 33244.33 36995.75 32230.31 38009.76
## 2012.673       37783.32 35908.85 39657.79 34895.51 40671.13
## 2012.692       36302.59 34400.57 38204.61 33372.34 39232.84
## 2012.712       34649.36 32774.57 36524.14 31761.06 37537.65
## 2012.731       35949.79 34074.91 37824.67 33061.34 38838.23
```

```
# plot of forecasted values
autoplot(training_1.4, series = "Training") +
  autolayer(forecast.lm.sales_1.4, alpha = 0.3, series = "Forecasts") +
  autolayer(validation_1.4, series = "Validation") +
  labs(title = "Dept. 4 LM Model Forecasted Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```



Department 5 Models:

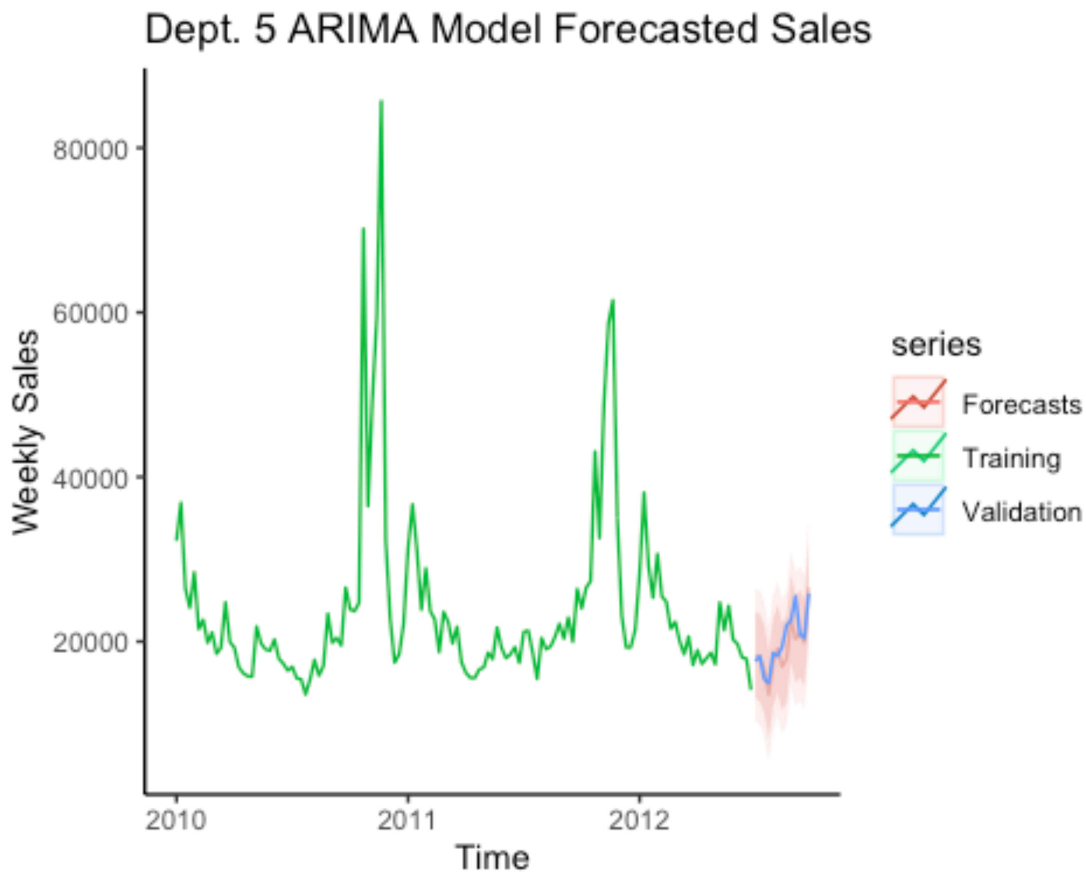
```
# Auto ARIMA model
#predictors.diff_1.1 <- diff(predictors_1.1)
#training.diff_1.1 <- diff(training_1.1)
AutoArima_1.5 <- auto.arima(training_1.5, xreg = predictors_1.5)
summary(AutoArima_1.5)
```

```
## Series: training_1.5
## Regression with ARIMA(0,0,0)(0,1,0)[52] errors
##
## Coefficients:
##          xreg
##       92.8756
## s.e.  77.8587
##
## sigma^2 = 22145163:  log likelihood = -769.79
## AIC=1543.57   AICc=1543.73   BIC=1548.28
##
## Training set error measures:
##                     ME     RMSE     MAE       MPE     MAPE      MASE
ACF1
## Training set -26.78433 3621.705 1518.52 0.8008909 6.045876 0.6010607 0.118
2823
```

```r
# ARIMA model parameters decided by the ACF and PACF plots above
arima_1.5 <- Arima(training_1.5, xreg = predictors_1.5, order = c(0, 0, 1), s
easonal = c(0, 1, 1))
summary(arima_1.5)
```

```
## Series: training_1.5
## Regression with ARIMA(0,0,1)(0,1,1)[52] errors
##
## Coefficients:
##          ma1     sma1      xreg
##       0.1317  -0.9999  108.6980
## s.e.  0.1146   0.3988   84.0944
##
## sigma^2 = 11266209:  log likelihood = -765.7
## AIC=1539.4   AICc=1539.95   BIC=1548.83
##
## Training set error measures:
##                     ME     RMSE      MAE       MPE     MAPE      MASE
## Training set -30.72939 2549.458 1047.808 0.4313792 4.128701 0.4147436
##                      ACF1
## Training set 0.0007677673
```

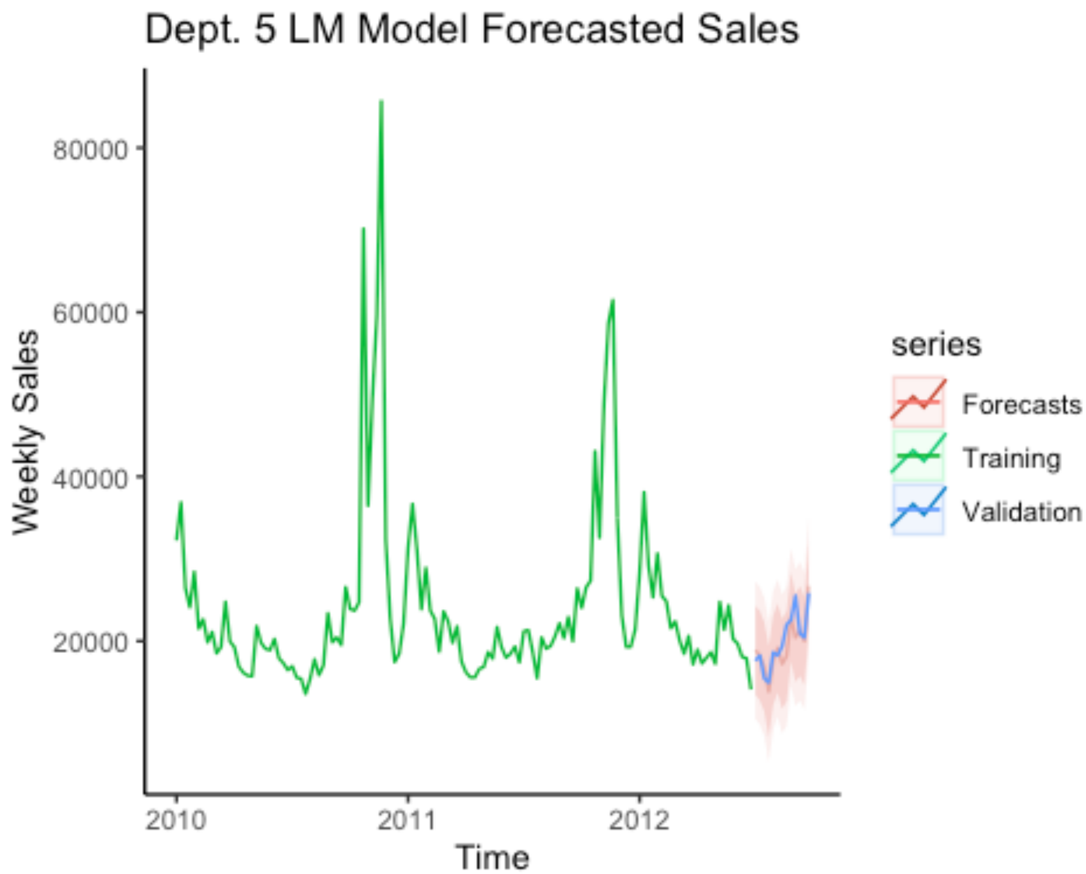
```r
# prediction on the arima
new.predictors_1.5 <- as.matrix(store_1.5["Temperature"][131:143,])
forecast.arima.sales_1.5 <- forecast(arima_1.5, xreg = new.predictors_1.5)

# plot of forecasted values
autoplot(training_1.5, series = "Training") +
  autolayer(forecast.arima.sales_1.5, alpha = 0.3, series = "Forecasts") +
  autolayer(validation_1.5, series = "Validation") +
  labs(title = "Dept. 5 ARIMA Model Forecasted Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```




```
# linear model
temp_1.5 <- store_1.5[1:130, 6]
linear_1.5 <- tslm(training_1.5 ~ trend + season + temp_1.5)
summary(linear_1.5)

##
## Call:
## tslm(formula = training_1.5 ~ trend + season + temp_1.5)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -13377.0   -886.6     15.5    891.0  13377.0
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   26252.056   4294.479   6.113 3.90e-08 ***
## trend             1.668      8.331   0.200 0.841829
## season2        7152.190   2791.206   2.562 0.012373 *
## season3       -1734.540   2744.683  -0.632 0.529307
## season4       -6949.676   2834.095  -2.452 0.016494 *
## season5       -2074.345   2835.419  -0.732 0.466673
## season6       -7924.841   2857.537  -2.773 0.006977 **
## season7       -8470.602   2966.780  -2.855 0.005543 **
```

```
## season8     -11974.589   3024.669  -3.959 0.000168 ***
## season9      -9704.905   3044.020  -3.188 0.002079 **
## season10    -12214.135   3246.441  -3.762 0.000329 ***
## season11    -13435.850   3283.503  -4.092 0.000106 ***
## season12    -10156.241   3251.261  -3.124 0.002527 **
## season13    -14397.020   3281.607  -4.387 3.65e-05 ***
## season14    -14654.494   3375.481  -4.341 4.31e-05 ***
## season15    -16553.780   3569.483  -4.638 1.44e-05 ***
## season16    -16282.540   3400.898  -4.788 8.14e-06 ***
## season17    -16472.916   3773.372  -4.366 3.95e-05 ***
## season18    -17064.564   3893.516  -4.383 3.71e-05 ***
## season19    -11940.486   3896.540  -3.064 0.003018 **
## season20    -14251.722   4056.233  -3.514 0.000748 ***
## season21    -12122.320   3982.622  -3.044 0.003208 **
## season22    -14517.873   4083.692  -3.555 0.000654 ***
## season23    -14560.970   4011.724  -3.630 0.000512 ***
## season24    -15747.692   4029.310  -3.908 0.000200 ***
## season25    -15742.118   4043.098  -3.894 0.000211 ***
## season26    -17983.213   4078.250  -4.410 3.36e-05 ***
## season27    -15391.881   4682.002  -3.287 0.001532 **
## season28    -15967.968   4647.293  -3.436 0.000960 ***
## season29    -17321.395   4609.059  -3.758 0.000334 ***
## season30    -19672.159   4502.224  -4.369 3.89e-05 ***
## season31    -16098.718   4375.787  -3.679 0.000435 ***
## season32    -14935.385   3969.670  -3.762 0.000329 ***
## season33    -16072.463   4169.885  -3.854 0.000241 ***
## season34    -14656.951   4021.764  -3.644 0.000488 ***
## season35    -10457.387   3884.768  -2.692 0.008735 **
## season36    -12288.445   3467.656  -3.544 0.000679 ***
## season37    -10974.505   3583.155  -3.063 0.003032 **
## season38    -12664.328   3472.340  -3.647 0.000483 ***
## season39     -6062.292   3524.298  -1.720 0.089476 .
## season40     -7547.394   3170.132  -2.381 0.019781 *
## season41     -6624.157   3228.052  -2.052 0.043608 *
## season42     -5490.243   3168.955  -1.733 0.087239 .
## season43     24608.153   3311.885   7.430 1.35e-10 ***
## season44      3702.409   3074.201   1.204 0.232192
## season45     18760.013   3078.248   6.094 4.21e-08 ***
## season46     28153.801   3083.527   9.130 7.52e-14 ***
## season47     42659.560   3079.931  13.851  < 2e-16 ***
## season48      3040.620   3074.367   0.989 0.325790
## season49     -7884.525   3074.523  -2.564 0.012305 *
## season50    -11820.242   3110.494  -3.800 0.000290 ***
## season51    -11961.807   3076.512  -3.888 0.000215 ***
## season52     -9081.440   3076.849  -2.952 0.004203 **
## temp_1.5        90.668     83.483   1.086 0.280878
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3361 on 76 degrees of freedom
```

```
## Multiple R-squared:  0.9443, Adjusted R-squared:  0.9054
## F-statistic: 24.29 on 53 and 76 DF,  p-value: < 2.2e-16

# calculating RMSE
sqrt(mean(linear_1.5$residuals^2))

## [1] 2569.963

# forecasting
temp.new_1.5 <- store_1.5[131:143, 6]
forecast.lm.sales_1.5 <- forecast(linear_1.5, temp.new_1.5, h = 13)

## Warning in forecast.lm(linear_1.5, temp.new_1.5, h = 13): newdata column n
ames
## not specified, defaulting to first variable required.

forecast.lm.sales_1.5

##          Point Forecast    Lo 80    Hi 80     Lo 95    Hi 95
## 2012.500       18886.15 13471.65 24300.65 10544.574 27227.73
## 2012.519       18215.63 12794.60 23636.66  9863.987 26567.26
## 2012.538       16845.73 11429.46 22262.00  8501.428 25190.04
## 2012.558       13844.73  8331.41 19358.05  5350.908 22338.56
## 2012.577       17676.43 12252.81 23100.06  9320.793 26032.07
## 2012.596       19156.05 13751.24 24560.87 10829.396 27482.71
## 2012.615       17205.53 11750.87 22660.19  8802.083 25608.98
## 2012.635       18160.30 12656.83 23663.78  9681.650 26638.96
## 2012.654       22924.59 17538.05 28311.13 14626.087 31223.09
## 2012.673       20412.46 15029.49 25795.44 12119.452 28705.48
## 2012.692       21223.96 15761.88 26686.04 12809.077 29638.84
## 2012.712       19987.33 14603.46 25371.20 11692.936 28281.73
## 2012.731       26698.93 21314.78 32083.08 18404.104 34993.76

# plot of forecasted values
autoplot(training_1.5, series = "Training") +
  autolayer(forecast.lm.sales_1.5, alpha = 0.3, series = "Forecasts") +
  autolayer(validation_1.5, series = "Validation") +
  labs(title = "Dept. 5 LM Model Forecasted Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```

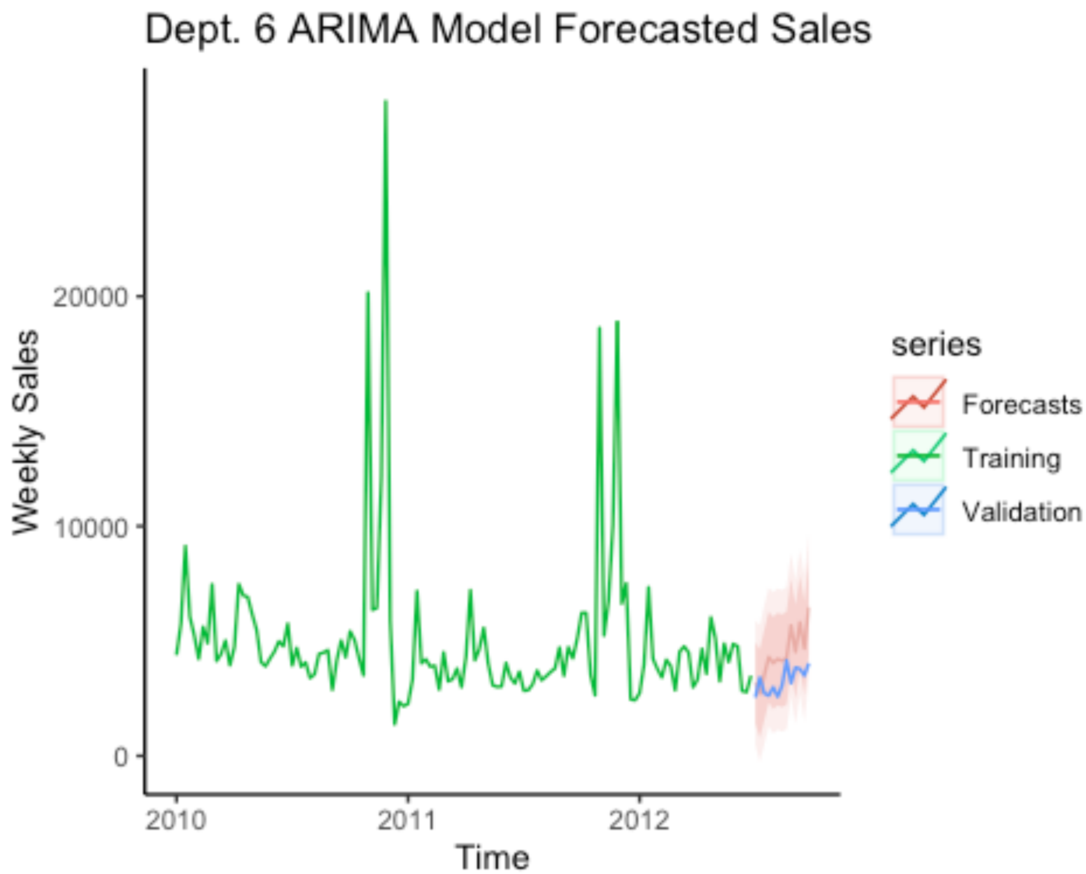


Department 6 Models:

```
# Auto ARIMA model
#predictors.diff_1.1 <- diff(predictors_1.1)
#training.diff_1.1 <- diff(training_1.1)
AutoArima_1.6 <- auto.arima(training_1.6, xreg = predictors_1.6)
summary(AutoArima_1.6)
```

```
## Series: training_1.6
## Regression with ARIMA(2,1,2)(0,1,0)[52] errors
##
## Coefficients:
##           ar1      ar2      ma1      ma2     xreg
##       -0.1649  -0.2878  -0.4236  -0.4366  11.5603
## s.e.   0.2015   0.1322   0.1967   0.1858  22.3813
##
## sigma^2 = 2340461:  log likelihood = -672.34
## AIC=1356.69   AICc=1357.89   BIC=1370.75
##
## Training set error measures:
##                      ME     RMSE      MAE      MPE     MAPE      MASE
ACF1
## Training set 154.5196 1138.532 570.3112 4.633932 12.25336 0.4759583 -0.025
36936
```

```r
# ARIMA model parameters decided by the ACF and PACF plots above
arima_1.6 <- Arima(training_1.6, xreg = predictors_1.6, order = c(2, 1, 3), s
easonal = c(0, 1, 1))
summary(arima_1.6)

## Series: training_1.6
## Regression with ARIMA(2,1,3)(0,1,1)[52] errors
##
## Coefficients:
##           ar1      ar2      ma1      ma2      ma3    sma1     xreg
##       -0.2377  -0.3328  -0.3276  -0.4555  -0.0637  0.9899   9.2830
## s.e.   0.6075   0.2364   0.6452   0.2001   0.4410  0.7164  17.9185
##
## sigma^2 = 1282534:  log likelihood = -670.8
## AIC=1357.61   AICc=1359.73   BIC=1376.36
##
## Training set error measures:
##                    ME     RMSE      MAE     MPE     MAPE      MASE
ACF1
## Training set 103.5397 831.0205 429.7156 3.21585 9.327156 0.3586229 -0.0221
7434

# prediction on the arima
new.predictors_1.6 <- as.matrix(store_1.6["Temperature"][131:143,])
forecast.arima.sales_1.6 <- forecast(arima_1.6, xreg = new.predictors_1.6)

# plot of forecasted values
autoplot(training_1.6, series = "Training") +
  autolayer(forecast.arima.sales_1.6, alpha = 0.3, series = "Forecasts") +
  autolayer(validation_1.6, series = "Validation") +
  labs(title = "Dept. 6 ARIMA Model Forecasted Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```




```
# linear model
temp_1.6 <- store_1.6[1:130, 6]
linear_1.6 <- tslm(training_1.6 ~ trend + season + temp_1.6)
summary(linear_1.6)

##
## Call:
## tslm(formula = training_1.6 ~ trend + season + temp_1.6)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4502.1  -434.2   -30.6   458.9  4502.1
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4084.306   1668.058   2.449 0.016649 *
## trend         -11.342      2.955  -3.838 0.000255 ***
## season2      1177.418   1019.618   1.155 0.251805
## season3      4681.709   1071.806   4.368 3.91e-05 ***
## season4      1623.899   1016.792   1.597 0.114398
## season5      1324.663    990.877   1.337 0.185257
## season6       766.134    990.904   0.773 0.441822
## season7      1520.571    991.199   1.534 0.129164
```

```
## season8       880.449    999.768   0.881 0.381282
## season9      1947.059   1007.274   1.933 0.056961 .
## season10      991.549   1010.139   0.982 0.329413
## season11     1238.483   1045.680   1.184 0.239953
## season12     1516.970   1053.202   1.440 0.153878
## season13      394.030   1046.827   0.376 0.707665
## season14     1217.848   1053.021   1.157 0.251088
## season15     3573.898   1072.876   3.331 0.001337 **
## season16     2066.149   1117.268   1.849 0.068305 .
## season17     3013.239   1078.666   2.793 0.006595 **
## season18     2828.987   1167.820   2.422 0.017798 *
## season19     1482.525   1198.973   1.236 0.220082
## season20     1259.080   1199.903   1.049 0.297357
## season21      935.381   1242.570   0.753 0.453908
## season22     1317.493   1222.948   1.077 0.284750
## season23     1751.058   1250.304   1.401 0.165431
## season24     1042.447   1231.028   0.847 0.399759
## season25      881.630   1235.904   0.713 0.477815
## season26     1633.229   1239.773   1.317 0.191676
## season27      436.707   1359.654   0.321 0.748948
## season28      876.009   1444.856   0.606 0.546125
## season29      606.625   1435.231   0.423 0.673731
## season30     1001.662   1424.666   0.703 0.484151
## season31      452.658   1395.233   0.324 0.746502
## season32      615.921   1360.959   0.453 0.652152
## season33     1116.495   1256.136   0.889 0.376897
## season34     1258.417   1306.767   0.963 0.338601
## season35     1763.110   1269.214   1.389 0.168849
## season36      281.012   1235.858   0.227 0.820737
## season37     1502.683   1146.407   1.311 0.193881
## season38     1717.305   1168.824   1.469 0.145890
## season39     1776.921   1147.433   1.549 0.125631
## season40     2920.902   1157.316   2.524 0.013694 *
## season41     2665.274   1108.697   2.404 0.018655 *
## season42      950.399   1112.280   0.854 0.395536
## season43      132.072   1108.885   0.119 0.905507
## season44    16500.650   1121.984  14.707  < 2e-16 ***
## season45     2830.276   1125.874   2.514 0.014058 *
## season46     3595.009   1150.073   3.126 0.002511 **
## season47     8114.825   1119.375   7.249 2.97e-10 ***
## season48    20786.495   1121.918  18.528  < 2e-16 ***
## season49     3314.327   1141.595   2.903 0.004832 **
## season50     1529.941   1129.396   1.355 0.179541
## season51     -537.147   1177.946  -0.456 0.649688
## season52     -593.571   1127.795  -0.526 0.600205
## temp_1.6       -6.326     27.581  -0.229 0.819219
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1213 on 76 degrees of freedom
```

```
## Multiple R-squared:  0.9258, Adjusted R-squared:  0.8741
## F-statistic: 17.89 on 53 and 76 DF,  p-value: < 2.2e-16
```

```
# calculating RMSE
sqrt(mean(linear_1.6$residuals^2))
```

```
## [1] 927.8138
```

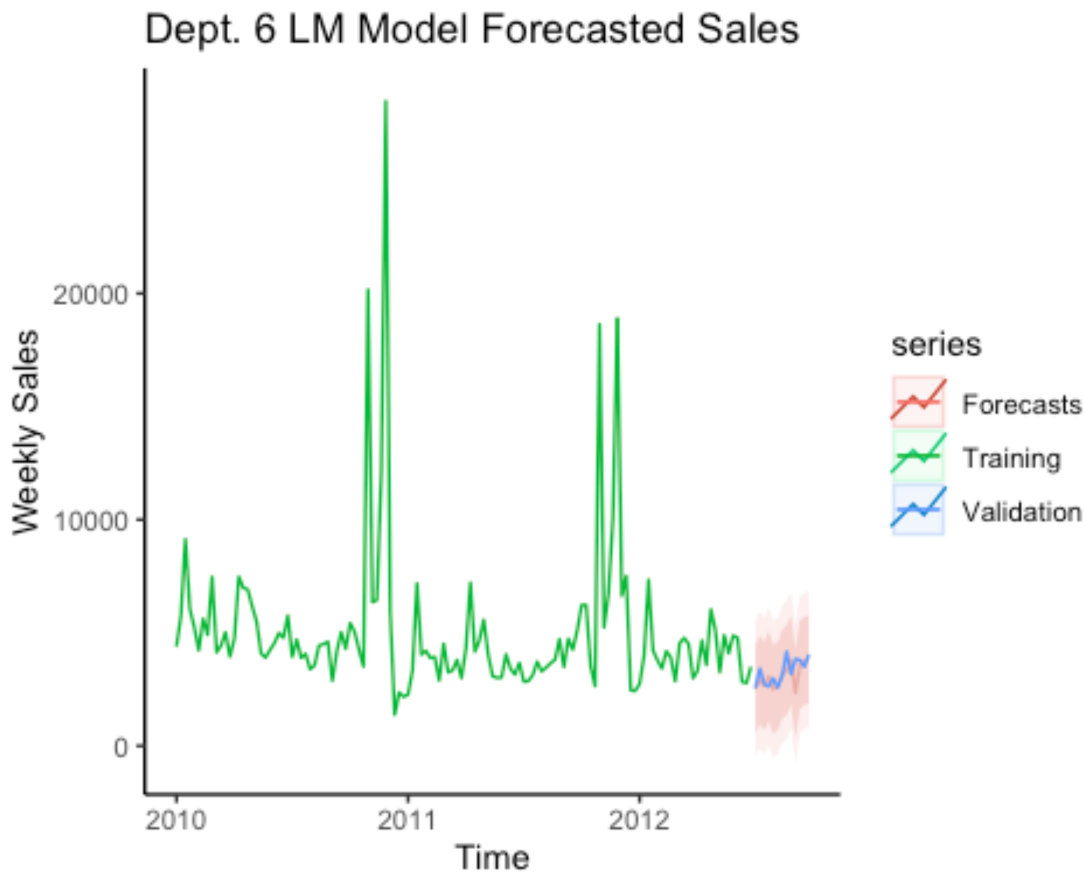
```
# forecasting
temp.new_1.6 <- store_1.6[131:143, 6]
forecast.lm.sales_1.6 <- forecast(linear_1.6, temp.new_1.6, h = 13)
```

```
## Warning in forecast.lm(linear_1.6, temp.new_1.6, h = 13): newdata column n
ames
## not specified, defaulting to first variable required.
```

```
forecast.lm.sales_1.6
```

```
##          Point Forecast      Lo 80    Hi 80       Lo 95    Hi 95
## 2012.500       2512.370  565.3117 4459.428 -487.267859 5512.008
## 2012.519       2918.507  967.0393 4869.975  -87.924223 5924.939
## 2012.538       2645.752  691.7229 4599.780 -364.624900 5656.128
## 2012.558       3074.927 1082.5071 5067.346    5.405221 6144.448
## 2012.577       2496.680  533.5769 4459.784 -527.676685 5521.038
## 2012.596       2626.652  681.6377 4571.666 -369.836839 5623.141
## 2012.615       3172.750 1223.9955 5121.505  170.498744 6175.002
## 2012.635       3335.591 1339.0003 5332.182  259.643543 6411.538
## 2012.654       3789.660 1841.1283 5738.192  787.752069 6791.568
## 2012.673       2343.852  374.5865 4313.116 -689.997973 5377.701
## 2012.692       3589.351 1636.7594 5541.943  581.188362 6597.514
## 2012.712       3761.130 1814.4667 5707.793  762.100723 6760.159
## 2012.731       3801.877 1858.3611 5745.392  807.696780 6796.056
```

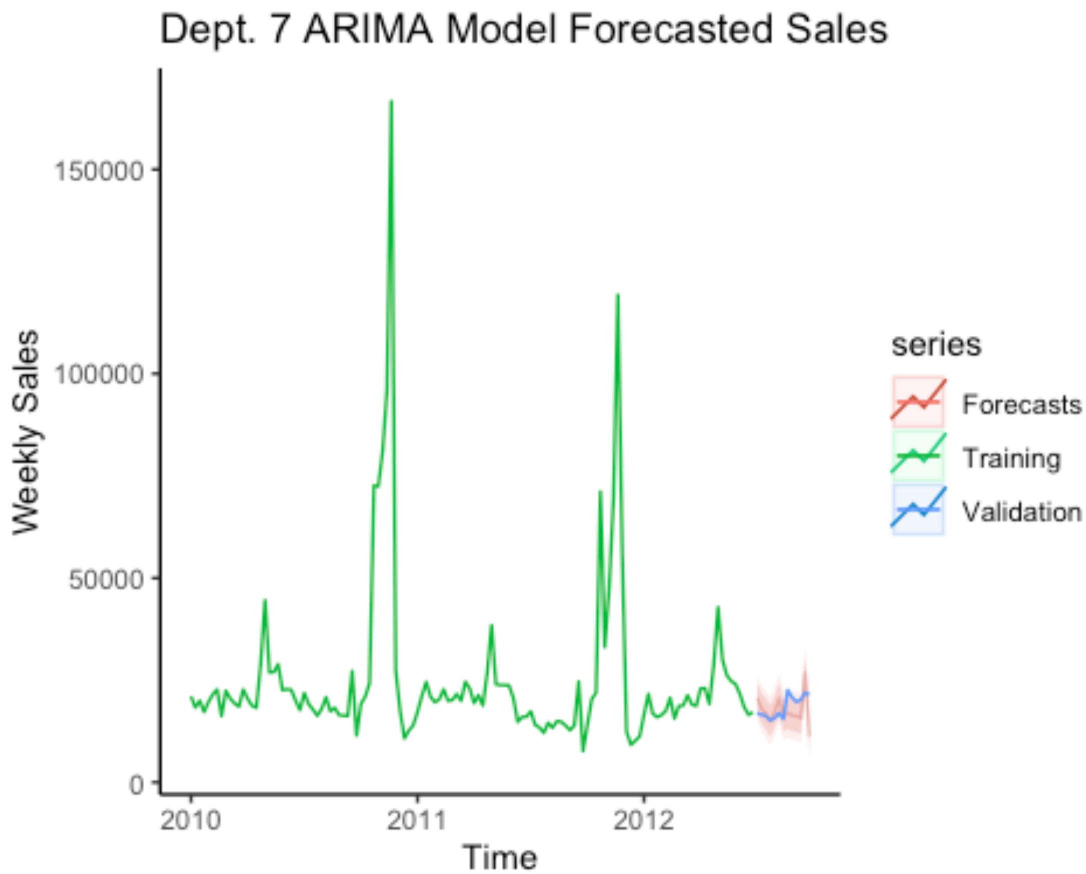
```
# plot of forecasted values
autoplot(training_1.6, series = "Training") +
  autolayer(forecast.lm.sales_1.6, alpha = 0.3, series = "Forecasts") +
  autolayer(validation_1.6, series = "Validation") +
  labs(title = "Dept. 6 LM Model Forecasted Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```



Department 7 Models:

```
# Auto ARIMA model
#predictors.diff_1.1 <- diff(predictors_1.1)
#training.diff_1.1 <- diff(training_1.1)
AutoArima_1.7 <- auto.arima(training_1.7, xreg = predictors_1.7)
summary(AutoArima_1.7)

## Series: training_1.7
## Regression with ARIMA(2,0,0)(0,1,0)[52] errors
##
## Coefficients:
##          ar1     ar2    drift      xreg
##       0.2126  0.2275  -55.677  151.4137
## s.e.  0.1091  0.1092   33.258  150.4872
##
## sigma^2 = 78277001:  log likelihood = -817.57
## AIC=1645.14   AICc=1645.98   BIC=1656.93
##
## Training set error measures:
##                      ME     RMSE      MAE        MPE     MAPE      MASE
## Training set 0.01804208 6675.154 2347.383 0.03687121 8.484803 0.4725356
##                      ACF1
## Training set -0.004579969
```

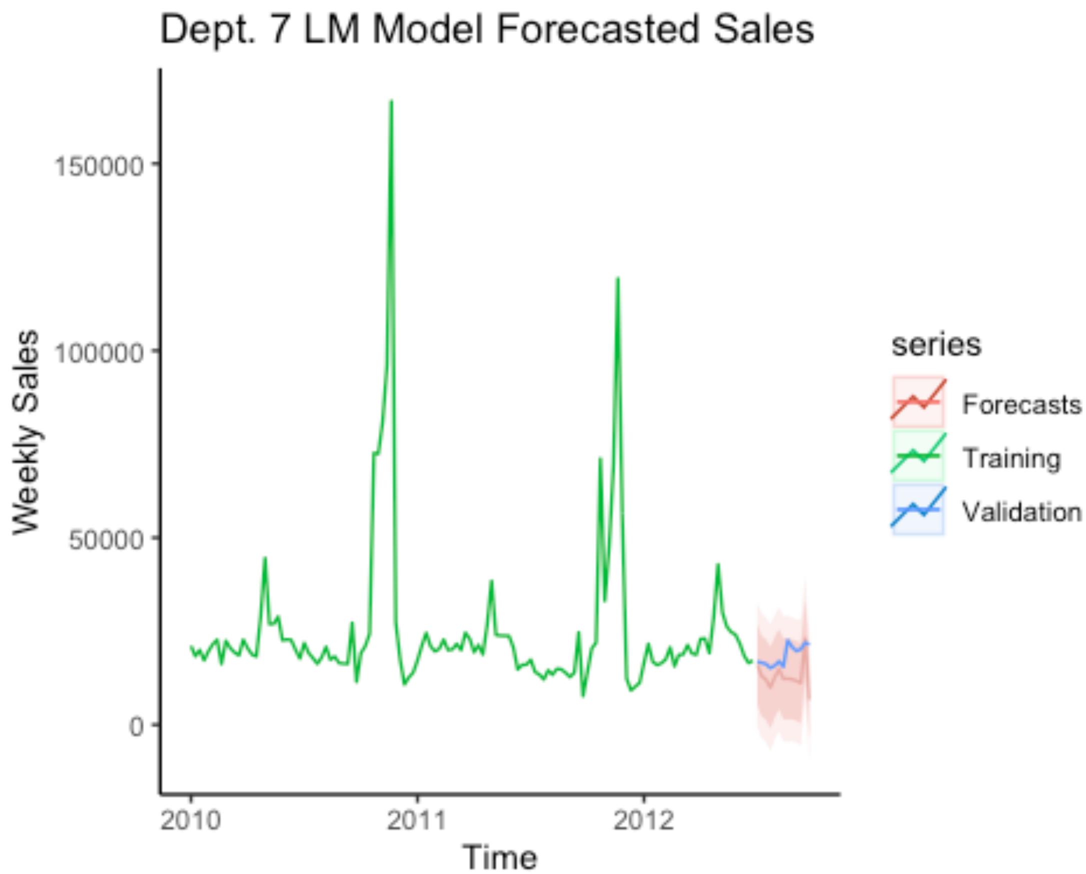
```r
# ARIMA model parameters decided by the ACF and PACF plots above
arima_1.7 <- Arima(training_1.7, xreg = predictors_1.7, order = c(2, 0, 1), s
easonal = c(1, 1, 0))
summary(arima_1.7)

## Series: training_1.7
## Regression with ARIMA(2,0,1)(1,1,0)[52] errors
##
## Coefficients:
##          ar1      ar2      ma1      sar1      xreg
##       0.8775  -0.0048  -0.6554  -0.9662   71.5952
## s.e.  0.0322   0.0310   0.1343   0.0034  116.6786
##
## sigma^2 = 7919706:  log likelihood = -798.2
## AIC=1608.41   AICc=1609.59   BIC=1622.55
##
## Training set error measures:
##                     ME     RMSE      MAE       MPE     MAPE      MASE
## Training set -269.0521 2108.843 956.0441 -1.274203 3.859567 0.1924547
##                    ACF1
## Training set -0.0304153

# prediction on the arima
new.predictors_1.7 <- as.matrix(store_1.7["Temperature"][131:143,])
forecast.arima.sales_1.7 <- forecast(arima_1.7, xreg = new.predictors_1.7)

# plot of forecasted values
autoplot(training_1.7, series = "Training") +
  autolayer(forecast.arima.sales_1.7, alpha = 0.3, series = "Forecasts") +
  autolayer(validation_1.7, series = "Validation") +
  labs(title = "Dept. 7 ARIMA Model Forecasted Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```




```r
# linear model
temp_1.7 <- store_1.7[1:130, 6]
linear_1.7 <- tslm(training_1.7 ~ trend + season + temp_1.7)
summary(linear_1.7)

##
## Call:
## tslm(formula = training_1.7 ~ trend + season + temp_1.7)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -22380.5  -1266.3     -7.6   1316.6  22380.5
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  14089.38    8518.75   1.654 0.102268
## trend          -39.42      16.52  -2.386 0.019543 *
## season2       2924.06    5536.78   0.528 0.598958
## season3       2180.79    5444.50   0.401 0.689875
## season4      -1264.63    5621.86  -0.225 0.822622
## season5       -710.95    5624.49  -0.126 0.899747
## season6        389.27    5668.36   0.069 0.945429
## season7       2167.93    5885.06   0.368 0.713615
```

```
## season8      -2798.45    5999.89  -0.466 0.642251
## season9        303.67    6038.28   0.050 0.960023
## season10      -436.34    6439.81  -0.068 0.946157
## season11      -625.68    6513.33  -0.096 0.923724
## season12        42.74    6449.37   0.007 0.994730
## season13       710.98    6509.57   0.109 0.913315
## season14       -36.94    6695.78  -0.006 0.995613
## season15      -394.75    7080.62  -0.056 0.955686
## season16     -2224.55    6746.20  -0.330 0.742497
## season17      6287.15    7485.06   0.840 0.403565
## season18     19851.95    7723.38   2.570 0.012115 *
## season19      4913.17    7729.38   0.636 0.526915
## season20      3293.69    8046.16   0.409 0.683435
## season21      3631.54    7900.14   0.460 0.647058
## season22      1090.79    8100.63   0.135 0.893240
## season23      -473.51    7957.87  -0.060 0.952708
## season24     -3555.66    7992.75  -0.445 0.657685
## season25     -4662.45    8020.10  -0.581 0.562728
## season26     -5241.60    8089.83  -0.648 0.518987
## season27     -4345.23    9287.47  -0.468 0.641224
## season28     -7217.76    9218.62  -0.783 0.436087
## season29     -8115.64    9142.77  -0.888 0.377526
## season30     -9253.14    8930.85  -1.036 0.303447
## season31     -6864.30    8680.04  -0.791 0.431513
## season32     -5052.57    7874.45  -0.642 0.523036
## season33     -6430.48    8271.60  -0.777 0.439326
## season34     -5676.36    7977.78  -0.712 0.478940
## season35     -6628.84    7706.03  -0.860 0.392376
## season36     -6003.12    6878.62  -0.873 0.385564
## season37     -5799.67    7107.74  -0.816 0.417070
## season38      5444.27    6887.92   0.790 0.431748
## season39    -11058.81    6990.98  -1.582 0.117833
## season40     -2500.82    6288.44  -0.398 0.691977
## season41      1303.27    6403.33   0.204 0.839265
## season42      4121.94    6286.11   0.656 0.513983
## season43     52185.33    6569.63   7.943 1.41e-11 ***
## season44     34873.79    6098.15   5.719 2.01e-07 ***
## season45     47073.83    6106.18   7.709 3.96e-11 ***
## season46     64816.17    6116.65  10.597  < 2e-16 ***
## season47    125079.87    6109.51  20.473  < 2e-16 ***
## season48     24532.85    6098.48   4.023 0.000135 ***
## season49     -2692.92    6098.79  -0.442 0.660068
## season50     -6723.14    6170.14  -1.090 0.279321
## season51     -6195.77    6102.73  -1.015 0.313210
## season52     -4983.03    6103.40  -0.816 0.416803
## temp_1.7       133.78     165.60   0.808 0.421696
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6667 on 76 degrees of freedom
```

```
## Multiple R-squared:  0.9378, Adjusted R-squared:  0.8944
## F-statistic: 21.62 on 53 and 76 DF,  p-value: < 2.2e-16

# calculating RMSE
sqrt(mean(linear_1.7$residuals^2))

## [1] 5097.913

# forecasting
temp.new_1.7 <- store_1.7[131:143, 6]
forecast.lm.sales_1.7 <- forecast(linear_1.7, temp.new_1.7, h = 13)

## Warning in forecast.lm(linear_1.7, temp.new_1.7, h = 13): newdata column n
ames
## not specified, defaulting to first variable required.

forecast.lm.sales_1.7

##          Point Forecast      Lo 80    Hi 80      Lo 95    Hi 95
## 2012.500      16099.828  5359.3380 26840.32  -446.9703 32646.63
## 2012.519      13046.075  2292.6335 23799.52 -3520.6764 29612.83
## 2012.538      12082.016  1338.0161 22826.02 -4470.1895 28634.22
## 2012.558       9943.207  -993.3099 20879.72 -6905.5904 26792.00
## 2012.577      12671.224  1912.6315 23429.82 -3903.4630 29245.91
## 2012.596      14907.752  4186.4772 25629.03 -1609.4436 31424.95
## 2012.615      12287.732  1467.5790 23107.89 -4381.7953 28957.26
## 2012.635      12320.148  1403.1627 23237.13 -4498.5589 29138.85
## 2012.654      12159.028  1474.0067 22844.05 -4302.3150 28620.37
## 2012.673      11737.954  1059.9975 22415.91 -4712.5050 28188.41
## 2012.692      11158.168   323.3000 21993.04 -5534.0292 27850.37
## 2012.712      23028.913 12349.1776 33708.65  6575.7133 39482.11
## 2012.731       6645.611 -4034.6792 17325.90 -9808.4433 23099.66

# plot of forecasted values
autoplot(training_1.7, series = "Training") +
  autolayer(forecast.lm.sales_1.7, alpha = 0.3, series = "Forecasts") +
  autolayer(validation_1.7, series = "Validation") +
  labs(title = "Dept. 7 LM Model Forecasted Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```

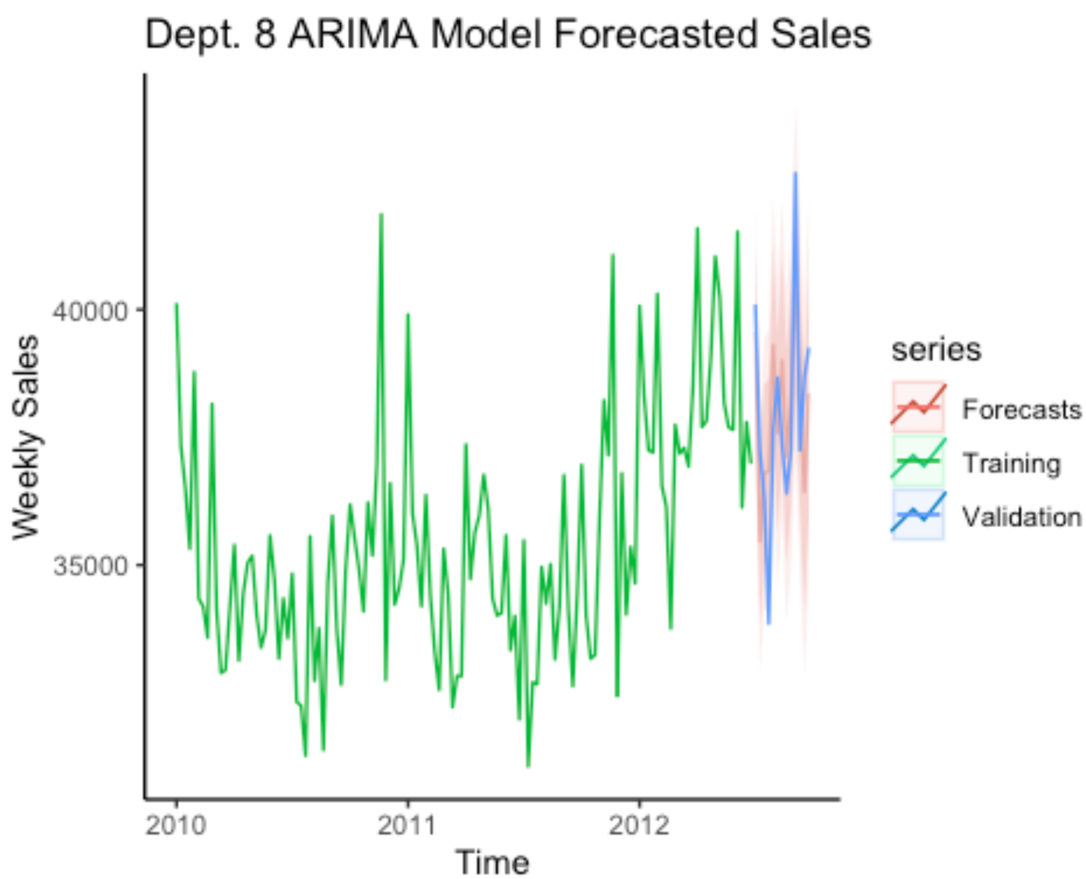


Department 8 Models:

```
# Auto ARIMA model
#predictors.diff_1.1 <- diff(predictors_1.1)
#training.diff_1.1 <- diff(training_1.1)
AutoArima_1.8 <- auto.arima(training_1.8, xreg = predictors_1.8)
summary(AutoArima_1.8)
```

```
## Series: training_1.8
## Regression with ARIMA(0,1,1)(0,1,0)[52] errors
##
## Coefficients:
##           ma1      xreg
##       -0.6632  -45.7139
## s.e.   0.0869   21.2363
##
## sigma^2 = 1482269:  log likelihood = -655.56
## AIC=1317.12   AICc=1317.44   BIC=1324.15
##
## Training set error measures:
##                     ME     RMSE      MAE       MPE     MAPE      MASE
## Training set 100.6768 924.7459 570.1565 0.2413113 1.587832 0.3226816
##                     ACF1
## Training set -0.01309687
```

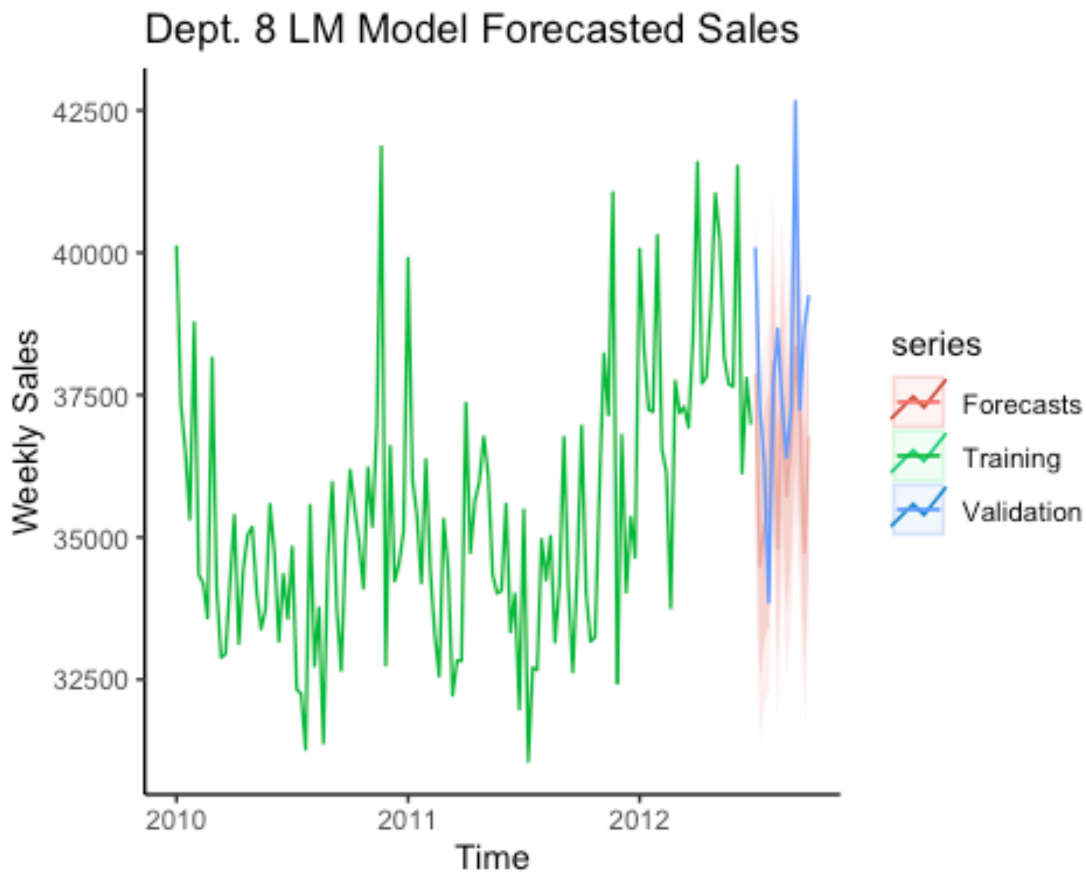
```r
# ARIMA model parameters decided by the ACF and PACF plots above
arima_1.8 <- Arima(training_1.8, xreg = predictors_1.8, order = c(1, 1, 1), s
easonal = c(0, 1, 1))
summary(arima_1.8)

## Series: training_1.8
## Regression with ARIMA(1,1,1)(0,1,1)[52] errors
##
## Coefficients:
##          ar1      ma1     sma1      xreg
##       0.0185  -0.6666  -0.1735  -46.9825
## s.e.  0.1677   0.1203   0.2143   22.3972
##
## sigma^2 = 1479111:  log likelihood = -655.21
## AIC=1320.42   AICc=1321.26   BIC=1332.14
##
## Training set error measures:
##                    ME     RMSE      MAE       MPE     MAPE      MASE
## Training set 100.0455 911.3604 569.2195 0.2388208 1.582977 0.3221513
##                     ACF1
## Training set -0.01536931

# prediction on the arima
new.predictors_1.8 <- as.matrix(store_1.8["Temperature"][131:143,])
forecast.arima.sales_1.8 <- forecast(arima_1.8, xreg = new.predictors_1.8)

# plot of forecasted values
autoplot(training_1.8, series = "Training") +
  autolayer(forecast.arima.sales_1.8, alpha = 0.3, series = "Forecasts") +
  autolayer(validation_1.8, series = "Validation") +
  labs(title = "Dept. 8 ARIMA Model Forecasted Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```




```
# linear model
temp_1.8 <- store_1.8[1:130, 6]
linear_1.8 <- tslm(training_1.8 ~ trend + season + temp_1.8)
summary(linear_1.8)

##
## Call:
## tslm(formula = training_1.8 ~ trend + season + temp_1.8)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2300.10  -621.60    54.33   598.13  2635.06
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 45194.55    1603.29  28.189  < 2e-16 ***
## trend          28.86       3.11   9.278 3.92e-14 ***
## season2     -3704.73    1042.06  -3.555 0.000654 ***
## season3     -3642.91    1024.69  -3.555 0.000654 ***
## season4     -3331.85    1058.07  -3.149 0.002342 **
## season5      -436.86    1058.57  -0.413 0.680995
## season6     -3700.53    1066.82  -3.469 0.000865 ***
## season7     -3722.11    1107.61  -3.360 0.001220 **
```

```
## season8     -4745.26    1129.22  -4.202 7.13e-05 ***
## season9      -921.12    1136.44  -0.811 0.420169
## season10    -2050.99    1212.02  -1.692 0.094701 .
## season11    -3079.56    1225.85  -2.512 0.014119 *
## season12    -3106.78    1213.82  -2.560 0.012468 *
## season13    -2097.61    1225.14  -1.712 0.090950 .
## season14     1098.35    1260.19   0.872 0.386187
## season15    -1313.48    1332.62  -0.986 0.327437
## season16    -1029.70    1269.68  -0.811 0.419903
## season17      667.18    1408.74   0.474 0.637144
## season18     1902.80    1453.59   1.309 0.194468
## season19     1012.36    1454.72   0.696 0.488608
## season20     -131.07    1514.34  -0.087 0.931254
## season21     -478.78    1486.86  -0.322 0.748331
## season22      347.94    1524.59   0.228 0.820092
## season23     1667.61    1497.72   1.113 0.269034
## season24    -1380.59    1504.29  -0.918 0.361642
## season25     -202.47    1509.44  -0.134 0.893648
## season26    -1354.26    1522.56  -0.889 0.376563
## season27     1149.12    1747.96   0.657 0.512906
## season28    -2419.53    1735.01  -1.395 0.167219
## season29    -1746.70    1720.73  -1.015 0.313282
## season30    -2526.01    1680.85  -1.503 0.137029
## season31      443.35    1633.64   0.271 0.786829
## season32    -2376.76    1482.02  -1.604 0.112923
## season33     -984.00    1556.77  -0.632 0.529231
## season34    -3511.45    1501.47  -2.339 0.021983 *
## season35    -1760.70    1450.33  -1.214 0.228506
## season36    -1146.35    1294.60  -0.885 0.378691
## season37    -3194.90    1337.72  -2.388 0.019409 *
## season38    -4898.41    1296.35  -3.779 0.000312 ***
## season39    -2711.29    1315.75  -2.061 0.042759 *
## season40    -2442.02    1183.53  -2.063 0.042495 *
## season41    -3920.80    1205.15  -3.253 0.001703 **
## season42    -5031.94    1183.09  -4.253 5.94e-05 ***
## season43    -4650.04    1236.45  -3.761 0.000331 ***
## season44    -4178.61    1147.71  -3.641 0.000494 ***
## season45    -4118.02    1149.22  -3.583 0.000596 ***
## season46    -3002.01    1151.19  -2.608 0.010967 *
## season47     1293.31    1149.85   1.125 0.264229
## season48    -8132.86    1147.77  -7.086 6.05e-10 ***
## season49    -3743.49    1147.83  -3.261 0.001661 **
## season50    -7288.73    1161.26  -6.277 1.95e-08 ***
## season51    -5485.81    1148.58  -4.776 8.50e-06 ***
## season52    -5593.19    1148.70  -4.869 5.95e-06 ***
## temp_1.8     -142.21      31.17  -4.563 1.90e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1255 on 76 degrees of freedom
```

```
## Multiple R-squared:  0.8318, Adjusted R-squared:  0.7146
## F-statistic: 7.094 on 53 and 76 DF,  p-value: 1.526e-14

# calculating RMSE
sqrt(mean(linear_1.8$residuals^2))

## [1] 959.4617

# forecasting
temp.new_1.8 <- store_1.8[131:143, 6]
forecast.lm.sales_1.8 <- forecast(linear_1.8, temp.new_1.8, h = 13)

## Warning in forecast.lm(linear_1.8, temp.new_1.8, h = 13): newdata column n
ames
## not specified, defaulting to first variable required.

forecast.lm.sales_1.8

##          Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## 2012.500       37878.22 35856.78 39899.65 34764.00 40992.44
## 2012.519       34489.16 32465.29 36513.03 31371.19 37607.14
## 2012.538       35219.29 33197.20 37241.38 32104.05 38334.52
## 2012.558       35491.32 33433.00 37549.65 32320.26 38662.38
## 2012.577       38087.09 36062.25 40111.93 34967.62 41206.55
## 2012.596       34802.37 32784.55 36820.18 31693.72 37911.01
## 2012.615       37502.44 35466.01 39538.87 34365.12 40639.76
## 2012.635       35729.12 33674.47 37783.77 32563.73 38894.52
## 2012.654       36625.61 34614.61 38636.60 33527.47 39723.74
## 2012.673       38339.65 36329.99 40349.32 35243.57 41435.74
## 2012.692       37110.64 35071.45 39149.84 33969.06 40252.23
## 2012.712       34727.78 32717.78 36737.78 31631.18 37824.38
## 2012.731       36774.53 34764.43 38784.63 33677.77 39871.30

# plot of forecasted values
autoplot(training_1.8, series = "Training") +
  autolayer(forecast.lm.sales_1.8, alpha = 0.3, series = "Forecasts") +
  autolayer(validation_1.8, series = "Validation") +
  labs(title = "Dept. 8 LM Model Forecasted Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```

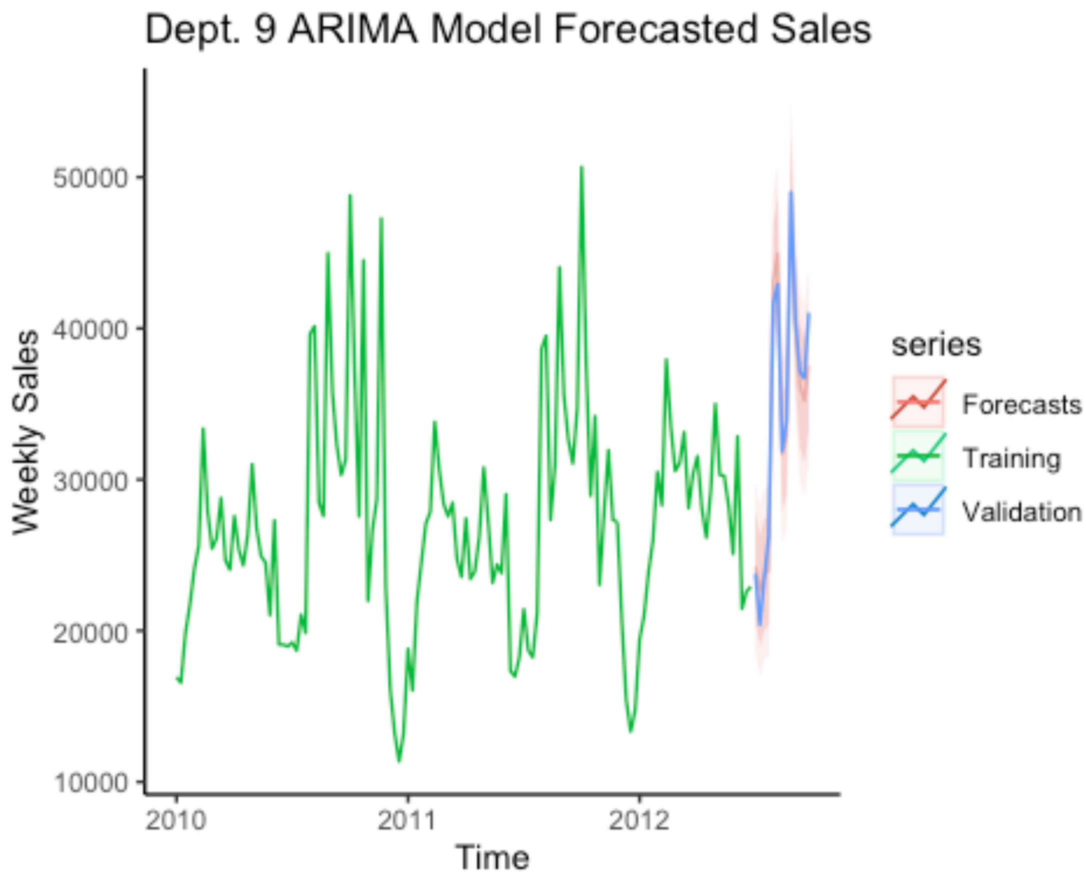


Department 9 Models:

```
# Auto ARIMA model
#predictors.diff_1.1 <- diff(predictors_1.1)
#training.diff_1.1 <- diff(training_1.1)
AutoArima_1.9 <- auto.arima(training_1.9, xreg = predictors_1.9)
summary(AutoArima_1.9)
```

```
## Series: training_1.9
## Regression with ARIMA(0,1,2)(0,1,0)[52] errors
##
## Coefficients:
##          ma1     ma2     xreg
##      -1.0764  0.2246  61.8295
## s.e.   0.1409  0.1443  53.6981
##
## sigma^2 = 11058829:  log likelihood = -732.94
## AIC=1473.88   AICc=1474.44   BIC=1483.26
##
## Training set error measures:
##                    ME     RMSE      MAE       MPE     MAPE      MASE
ACF1
## Training set 168.8795 2508.989 1180.516 0.7077426 4.671497 0.4613915 0.027
93121
```

```r
# ARIMA model parameters decided by the ACF and PACF plots above
arima_1.9 <- Arima(training_1.9, xreg = predictors_1.9, order = c(0, 1, 4), s
easonal = c(0, 1, 1))
summary(arima_1.9)

## Series: training_1.9
## Regression with ARIMA(0,1,4)(0,1,1)[52] errors
##
## Coefficients:
##           ma1     ma2     ma3     ma4     sma1      xreg
##       -1.0311  0.0685  0.0985  0.0415  -0.9998  103.8517
## s.e.   0.1324  0.1609  0.1596  0.1300   0.3516   58.4138
##
## sigma^2 = 5412065:  log likelihood = -726.94
## AIC=1467.88   AICc=1469.51   BIC=1484.29
##
## Training set error measures:
##                     ME    RMSE      MAE       MPE     MAPE      MASE
ACF1
## Training set 95.69437 1719.25 780.5302 0.4228348 3.092106 0.3050615 -0.023
48425

# prediction on the arima
new.predictors_1.9 <- as.matrix(store_1.9["Temperature"][131:143,])
forecast.arima.sales_1.9 <- forecast(arima_1.9, xreg = new.predictors_1.9)

# plot of forecasted values
autoplot(training_1.9, series = "Training") +
  autolayer(forecast.arima.sales_1.9, alpha = 0.3, series = "Forecasts") +
  autolayer(validation_1.9, series = "Validation") +
  labs(title = "Dept. 9 ARIMA Model Forecasted Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```




```
# linear model
temp_1.9 <- store_1.9[1:130, 6]
linear_1.9 <- tslm(training_1.9 ~ trend + season + temp_1.9)
summary(linear_1.9)

##
## Call:
## tslm(formula = training_1.9 ~ trend + season + temp_1.9)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10700.7   -798.1    183.8    786.4  10700.7
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 15462.445   3106.385   4.978 3.91e-06 ***
## trend          30.412      6.026   5.047 2.98e-06 ***
## season2      -376.184   2019.002  -0.186 0.852689
## season3      3409.244   1985.349   1.717 0.090015 .
## season4      5338.135   2050.025   2.604 0.011079 *
## season5      8467.089   2050.983   4.128 9.28e-05 ***
## season6      8445.275   2066.982   4.086 0.000108 ***
## season7     16095.232   2146.002   7.500 9.91e-11 ***
```

```
## season8     11673.255   2187.875   5.335 9.51e-07 ***
## season9      9054.041   2201.873   4.112 9.83e-05 ***
## season10     9010.102   2348.293   3.837 0.000256 ***
## season11    10832.643   2375.102   4.561 1.92e-05 ***
## season12     6524.248   2351.780   2.774 0.006960 **
## season13     6627.323   2373.730   2.792 0.006623 **
## season14     9410.471   2441.633   3.854 0.000241 ***
## season15     6033.755   2581.964   2.337 0.022081 *
## season16     5275.828   2460.018   2.145 0.035178 *
## season17     7541.158   2729.445   2.763 0.007182 **
## season18    12433.061   2816.351   4.415 3.30e-05 ***
## season19     8243.079   2818.538   2.925 0.004544 **
## season20     6145.443   2934.051   2.095 0.039546 *
## season21     5744.411   2880.805   1.994 0.049736 *
## season22     3258.543   2953.913   1.103 0.273453
## season23     9676.900   2901.856   3.335 0.001322 **
## season24     -785.588   2914.576  -0.270 0.788246
## season25     -574.715   2924.550  -0.197 0.844732
## season26     -128.758   2949.977  -0.044 0.965300
## season27      757.804   3386.697   0.224 0.823545
## season28     -886.793   3361.591  -0.264 0.792647
## season29       56.452   3333.934   0.017 0.986535
## season30      920.468   3256.656   0.283 0.778219
## season31    19595.397   3165.199   6.191 2.81e-08 ***
## season32    20441.449   2871.436   7.119 5.24e-10 ***
## season33     8391.734   3016.261   2.782 0.006807 **
## season34     9814.350   2909.118   3.374 0.001170 **
## season35    25079.505   2810.023   8.925 1.86e-13 ***
## season36    16511.082   2508.307   6.583 5.31e-09 ***
## season37    13147.554   2591.853   5.073 2.70e-06 ***
## season38    11411.613   2511.696   4.543 2.05e-05 ***
## season39    13666.659   2549.279   5.361 8.59e-07 ***
## season40    30679.916   2293.095  13.379  < 2e-16 ***
## season41    18263.563   2334.991   7.822 2.41e-11 ***
## season42     9131.959   2292.244   3.984 0.000154 ***
## season43    20051.901   2395.632   8.370 2.15e-12 ***
## season44     3550.855   2223.704   1.597 0.114456
## season45     8302.005   2226.632   3.729 0.000369 ***
## season46    11265.821   2230.450   5.051 2.94e-06 ***
## season47    18223.912   2227.849   8.180 4.98e-12 ***
## season48     6029.357   2223.824   2.711 0.008284 **
## season49     -681.076   2223.937  -0.306 0.760253
## season50    -4628.210   2249.957  -2.057 0.043116 *
## season51    -6828.265   2225.376  -3.068 0.002982 **
## season52    -5231.000   2225.619  -2.350 0.021351 *
## temp_1.9       27.884     60.387   0.462 0.645570
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2431 on 76 degrees of freedom
```

```
## Multiple R-squared:  0.9378, Adjusted R-squared:  0.8943
## F-statistic:  21.6 on 53 and 76 DF,  p-value: < 2.2e-16
```

```
# calculating RMSE
sqrt(mean(linear_1.9$residuals^2))
```

```
## [1] 1858.967
```

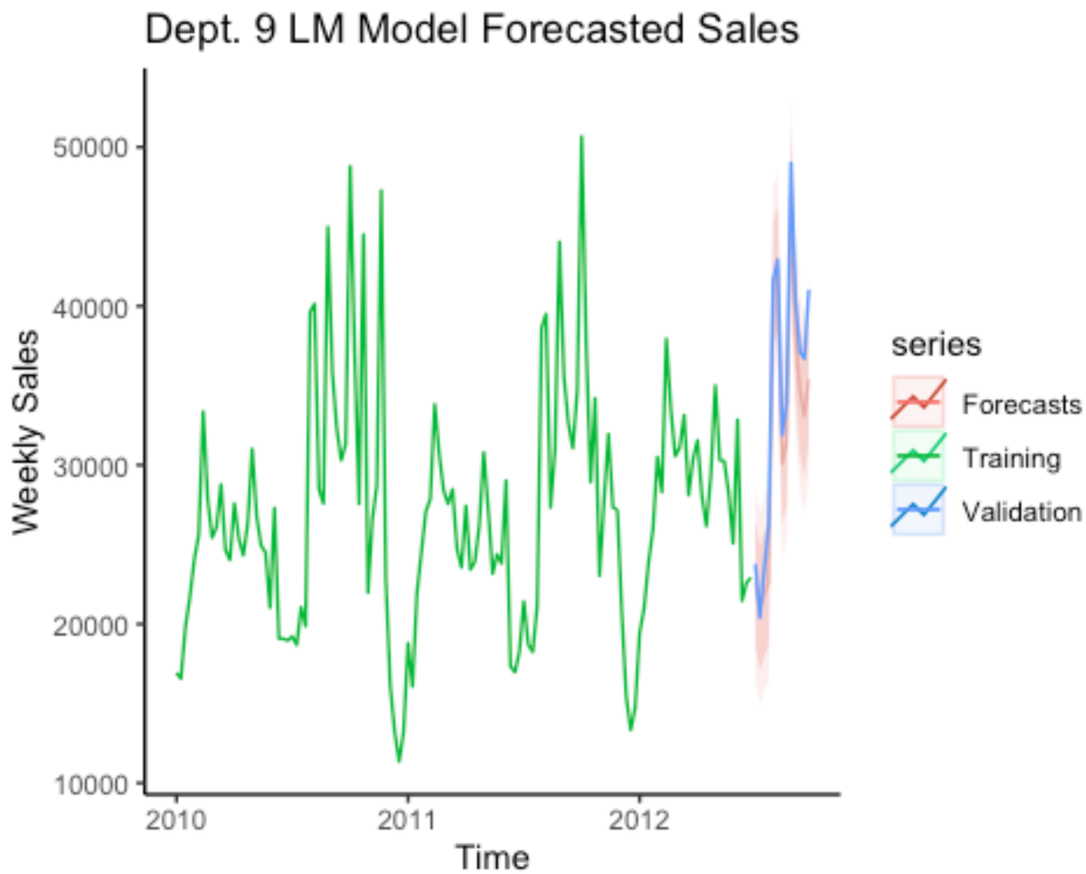
```
# forecasting
temp.new_1.9 <- store_1.9[131:143, 6]
forecast.lm.sales_1.9 <- forecast(linear_1.9, temp.new_1.9, h = 13)
```

```
## Warning in forecast.lm(linear_1.9, temp.new_1.9, h = 13): newdata column n
ames
## not specified, defaulting to first variable required.
```

```
forecast.lm.sales_1.9
```

```
##          Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## 2012.500       22605.31 18688.76 26521.86 16571.48 28639.14
## 2012.519       20961.57 17040.30 24882.84 14920.46 27002.67
## 2012.538       21929.65 18011.82 25847.47 15893.85 27965.45
## 2012.558       22623.59 18635.56 26611.62 16479.63 28767.54
## 2012.577       41407.84 37484.69 45330.99 35363.84 47451.84
## 2012.596       42381.06 38471.52 46290.60 36358.03 48404.10
## 2012.615       30111.08 26165.48 34056.68 24032.50 36189.66
## 2012.635       31421.90 27440.99 35402.80 25288.92 37554.88
## 2012.654       46890.63 42994.31 50786.94 40887.96 52893.29
## 2012.673       38142.65 34248.90 42036.39 32143.95 44141.35
## 2012.692       34654.49 30703.53 38605.45 28567.64 40741.34
## 2012.712       33087.83 29193.44 36982.22 27088.13 39087.53
## 2012.731       35406.47 31511.87 39301.06 29406.46 41406.48
```

```
# plot of forecasted values
autoplot(training_1.9, series = "Training") +
  autolayer(forecast.lm.sales_1.9, alpha = 0.3, series = "Forecasts") +
  autolayer(validation_1.9, series = "Validation") +
  labs(title = "Dept. 9 LM Model Forecasted Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```



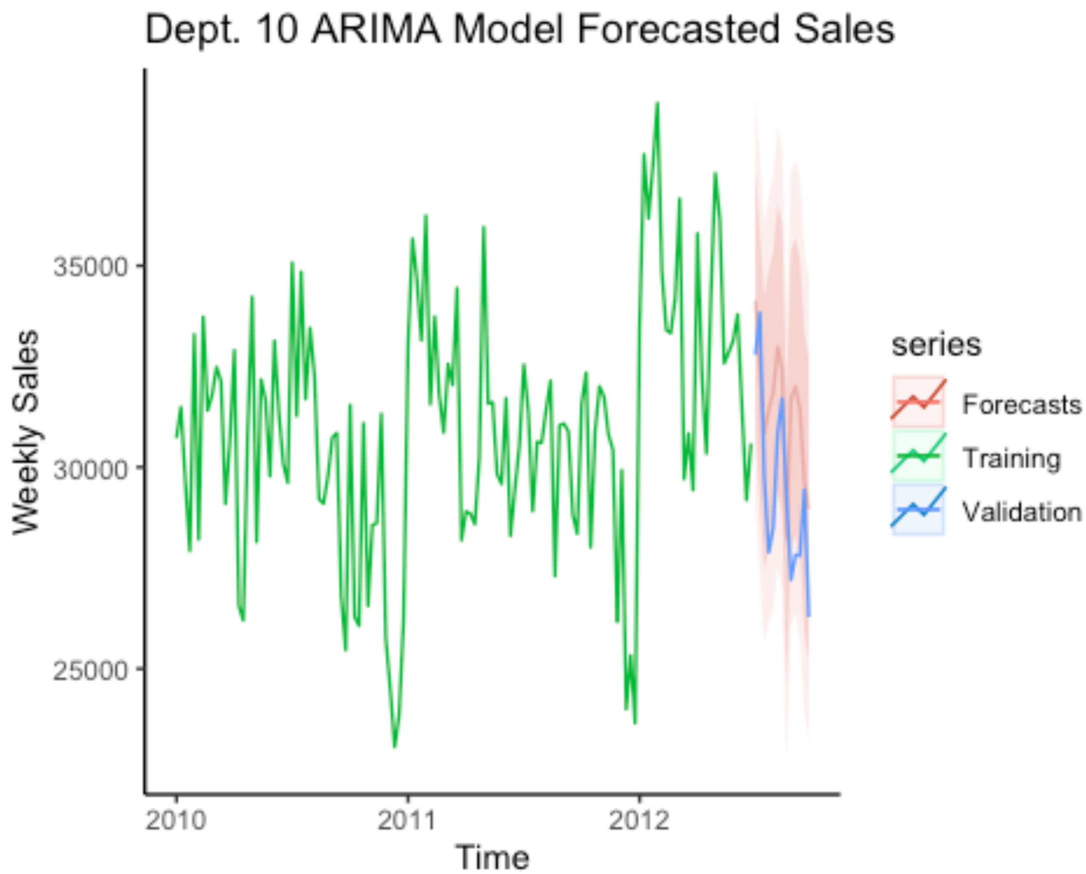
Department 10 Models:

```
# Auto ARIMA model
#predictors.diff_1.1 <- diff(predictors_1.1)
#training.diff_1.1 <- diff(training_1.1)
AutoArima_1.10 <- auto.arima(training_1.10, xreg = predictors_1.10)
summary(AutoArima_1.10)

## Series: training_1.10
## Regression with ARIMA(1,0,1)(0,1,0)[52] errors
##
## Coefficients:
##          ar1      ma1     xreg
##       0.9217  -0.6960  61.8860
## s.e.  0.0825   0.1521  44.2999
##
## sigma^2 = 6213341:  log likelihood = -719.44
## AIC=1446.89   AICc=1447.43   BIC=1456.31
##
## Training set error measures:
##                    ME     RMSE      MAE      MPE     MAPE      MASE
ACF1
## Training set 191.0152 1893.309 1214.645 0.413074 3.881143 0.5353974 0.0435
4789
```

```r
# ARIMA model parameters decided by the ACF and PACF plots above
arima_1.10 <- Arima(training_1.10, xreg = predictors_1.10, order = c(1, 0, 4)
, seasonal = c(0, 1, 2))
summary(arima_1.10)
```

```
## Series: training_1.10
## Regression with ARIMA(1,0,4)(0,1,2)[52] errors
##
## Coefficients:
##          ar1      ma1      ma2     ma3     ma4     sma1    sma2     xreg
##       0.9329  -0.6562  -0.1107  0.0534  0.0195  -0.1529  0.8665  50.4281
## s.e.  0.2122   0.2350   0.1692  0.1884  0.2155   0.3619  3.4224  47.7553
##
## sigma^2 = 3694778:  log likelihood = -718.56
## AIC=1455.12   AICc=1457.77   BIC=1476.33
##
## Training set error measures:
##                    ME     RMSE      MAE       MPE     MAPE     MASE
## Training set 130.8027 1410.495 908.4041 0.2734734 2.898582 0.400411
##                      ACF1
## Training set -0.006280305
```

```r
# prediction on the arima
new.predictors_1.10 <- as.matrix(store_1.10["Temperature"][131:143,])
forecast.arima.sales_1.10 <- forecast(arima_1.10, xreg = new.predictors_1.10)

# plot of forecasted values
autoplot(training_1.10, series = "Training") +
  autolayer(forecast.arima.sales_1.10, alpha = 0.3, series = "Forecasts") +
  autolayer(validation_1.10, series = "Validation") +
  labs(title = "Dept. 10 ARIMA Model Forecasted Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```




```r
# linear model
temp_1.10 <- store_1.10[1:130, 6]
linear_1.10 <- tslm(training_1.10 ~ trend + season + temp_1.10)
summary(linear_1.10)

##
## Call:
## tslm(formula = training_1.10 ~ trend + season + temp_1.10)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3665.5 -1000.0    32.3   974.1  3665.5
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 29344.57    2443.58  12.009  < 2e-16 ***
## trend          24.34       4.74   5.136 2.10e-06 ***
## season2      2756.98    1588.21   1.736 0.086633 .
## season3       992.70    1561.74   0.636 0.526920
## season4        32.83    1612.61   0.020 0.983809
## season5      3339.90    1613.37   2.070 0.041836 *
## season6     -1374.39    1625.95  -0.845 0.400607
## season7       531.75    1688.11   0.315 0.753627
```

```
## season8     -1005.98    1721.05  -0.585 0.560606
## season9      -942.09    1732.06  -0.544 0.588094
## season10      458.42    1847.24   0.248 0.804676
## season11    -2208.44    1868.33  -1.182 0.240875
## season12    -2042.89    1849.98  -1.104 0.272957
## season13    -4122.96    1867.25  -2.208 0.030257 *
## season14    -1118.00    1920.66  -0.582 0.562229
## season15    -4593.91    2031.05  -2.262 0.026566 *
## season16    -5349.07    1935.13  -2.764 0.007156 **
## season17    -2236.41    2147.07  -1.042 0.300894
## season18     1709.11    2215.43   0.771 0.442827
## season19    -2184.37    2217.15  -0.985 0.327644
## season20    -2148.38    2308.02  -0.931 0.354888
## season21    -2800.39    2266.13  -1.236 0.220354
## season22    -3505.71    2323.64  -1.509 0.135518
## season23    -1437.66    2282.69  -0.630 0.530707
## season24    -3998.04    2292.70  -1.744 0.085235 .
## season25    -4814.06    2300.54  -2.093 0.039725 *
## season26    -4192.70    2320.54  -1.807 0.074756 .
## season27     -228.64    2664.08  -0.086 0.931833
## season28    -2718.50    2644.33  -1.028 0.307188
## season29    -2169.42    2622.58  -0.827 0.410708
## season30    -2846.07    2561.79  -1.111 0.270083
## season31    -1921.17    2489.84  -0.772 0.442743
## season32    -1867.73    2258.76  -0.827 0.410892
## season33    -3188.34    2372.68  -1.344 0.183021
## season34    -5593.10    2288.40  -2.444 0.016839 *
## season35    -3317.26    2210.45  -1.501 0.137572
## season36    -2484.03    1973.11  -1.259 0.211906
## season37    -2657.24    2038.83  -1.303 0.196403
## season38    -5630.80    1975.78  -2.850 0.005626 **
## season39    -6615.57    2005.34  -3.299 0.001478 **
## season40    -1551.89    1803.82  -0.860 0.392309
## season41    -3920.25    1836.78  -2.134 0.036042 *
## season42    -6125.63    1803.15  -3.397 0.001086 **
## season43    -2415.53    1884.48  -1.282 0.203809
## season44    -3637.11    1749.23  -2.079 0.040967 *
## season45    -2634.48    1751.54  -1.504 0.136702
## season46    -3317.88    1754.54  -1.891 0.062432 .
## season47    -2152.74    1752.50  -1.228 0.223092
## season48    -6961.81    1749.33  -3.980 0.000156 ***
## season49    -5810.18    1749.42  -3.321 0.001379 **
## season50    -9266.90    1769.89  -5.236 1.42e-06 ***
## season51    -8497.07    1750.55  -4.854 6.31e-06 ***
## season52    -8175.82    1750.74  -4.670 1.27e-05 ***
## temp_1.10      38.11      47.50   0.802 0.424904
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1913 on 76 degrees of freedom
```

```
## Multiple R-squared:  0.7765, Adjusted R-squared:  0.6206
## F-statistic: 4.982 on 53 and 76 DF,  p-value: 1.4e-10
```

```
# calculating RMSE
sqrt(mean(linear_1.10$residuals^2))
```

```
## [1] 1462.321
```

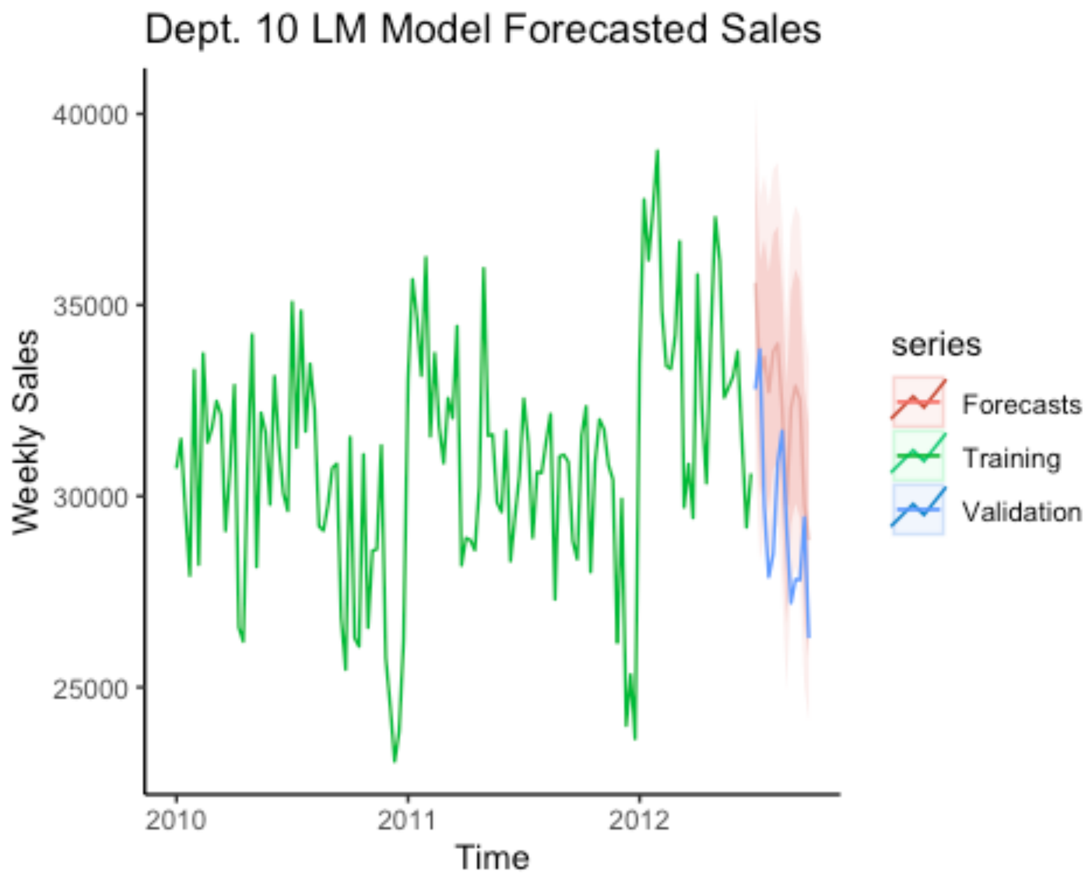
```
# forecasting
temp.new_1.10 <- store_1.10[131:143, 6]
forecast.lm.sales_1.10 <- forecast(linear_1.10, temp.new_1.10, h = 13)
```

```
## Warning in forecast.lm(linear_1.10, temp.new_1.10, h = 13): newdata column
names
## not specified, defaulting to first variable required.
```

```
forecast.lm.sales_1.10
```

```
##          Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## 2012.500       35586.54 32505.66 38667.41 30840.14 40332.94
## 2012.519       33080.62 29996.03 36165.22 28328.50 37832.75
## 2012.538       33646.43 30564.54 36728.31 28898.48 38394.38
## 2012.558       32720.11 29583.01 35857.22 27887.09 37553.14
## 2012.577       33777.21 30691.14 36863.28 29022.81 38531.61
## 2012.596       33987.23 30911.87 37062.60 29249.32 38725.14
## 2012.615       32348.36 29244.63 35452.09 27566.76 37129.96
## 2012.635       29773.59 26642.09 32905.09 24949.19 34597.98
## 2012.654       32310.43 29245.47 35375.40 27588.55 37032.32
## 2012.673       32881.05 29818.11 35943.99 28162.29 37599.82
## 2012.692       32520.30 29412.35 35628.25 27732.19 37308.41
## 2012.712       29760.86 26697.42 32824.31 25041.31 34480.42
## 2012.731       28845.79 25782.18 31909.39 24125.99 33565.58
```

```
# plot of forecasted values
autoplot(training_1.10, series = "Training") +
  autolayer(forecast.lm.sales_1.10, alpha = 0.3, series = "Forecasts") +
  autolayer(validation_1.10, series = "Validation") +
  labs(title = "Dept. 10 LM Model Forecasted Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```



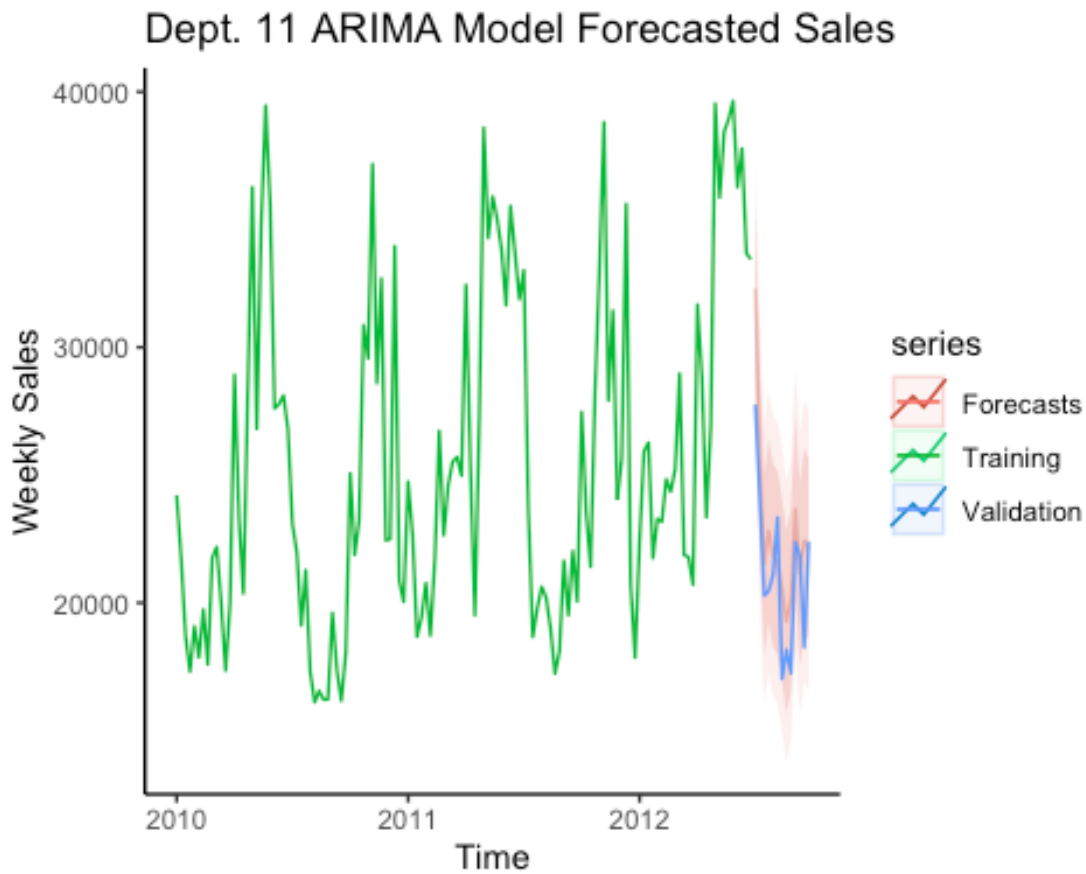
Department 11 Models:

```
# Auto ARIMA model
#predictors.diff_1.1 <- diff(predictors_1.1)
#training.diff_1.1 <- diff(training_1.1)
AutoArima_1.11 <- auto.arima(training_1.11, xreg = predictors_1.11)
summary(AutoArima_1.11)

## Series: training_1.11
## Regression with ARIMA(0,0,1)(0,1,0)[52] errors
##
## Coefficients:
##          ma1    drift     xreg
##       0.5171  36.4454  55.6796
## s.e.  0.0842   8.7526  46.4534
##
## sigma^2 = 7200287:  log likelihood = -725.1
## AIC=1458.19   AICc=1458.74   BIC=1467.62
##
## Training set error measures:
##                    ME     RMSE      MAE        MPE     MAPE      MASE
ACF1
## Training set 12.50009 2038.139 1276.635 -0.3792063 4.934208 0.4369941 0.03
83231
```

```r
# ARIMA model parameters decided by the ACF and PACF plots above
arima_1.11 <- Arima(training_1.11, xreg = predictors_1.11, order = c(0, 1, 2)
, seasonal = c(1, 1, 0))
summary(arima_1.11)
```

```
## Series: training_1.11
## Regression with ARIMA(0,1,2)(1,1,0)[52] errors
##
## Coefficients:
##           ma1      ma2     sar1     xreg
##       -0.5053  -0.4947  -0.4191  41.2856
## s.e.   0.1068   0.0927   0.1521  50.0273
##
## sigma^2 = 6034169:  log likelihood = -715.35
## AIC=1440.7   AICc=1441.54   BIC=1452.41
##
## Training set error measures:
##                     ME     RMSE      MAE        MPE     MAPE      MASE
## Training set -5.165186 1840.766 1141.668 -0.4183546 4.404359 0.3907944
##                   ACF1
## Training set 0.02744617
```

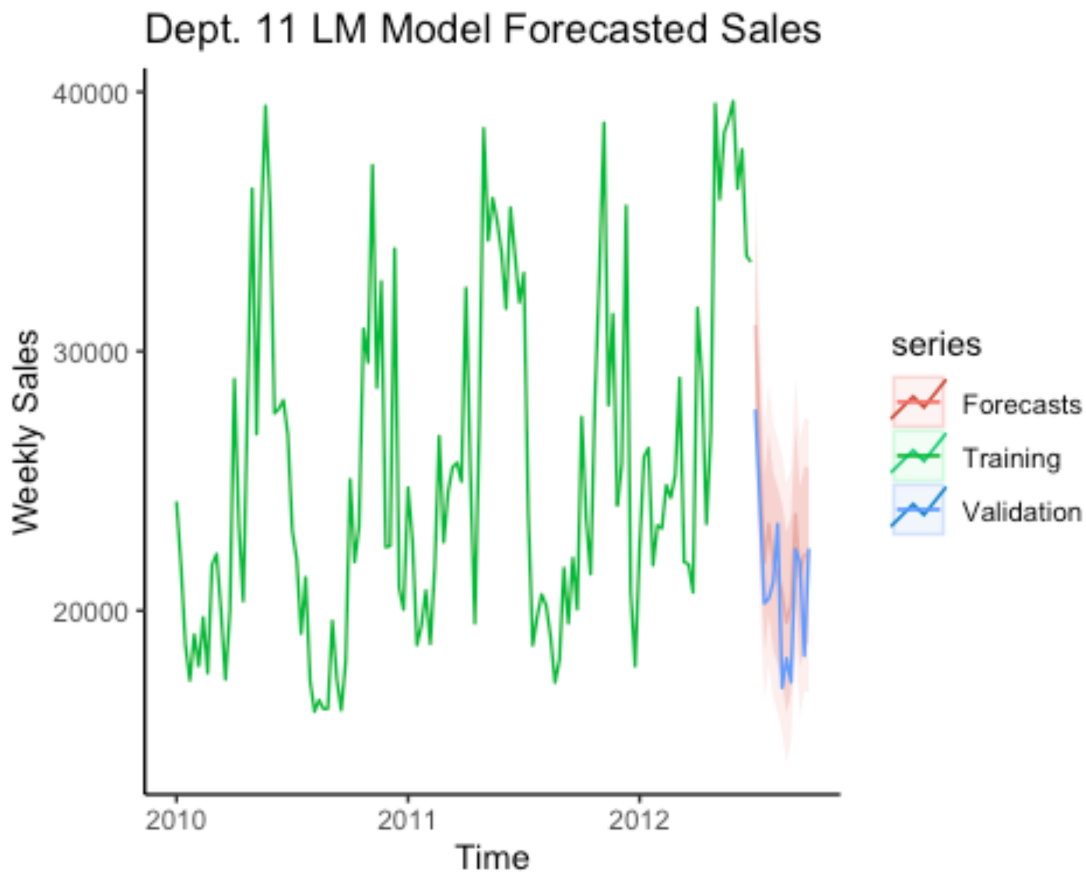
```r
# prediction on the arima
new.predictors_1.11 <- as.matrix(store_1.11["Temperature"][131:143,])
forecast.arima.sales_1.11 <- forecast(arima_1.11, xreg = new.predictors_1.11)

# plot of forecasted values
autoplot(training_1.11, series = "Training") +
  autolayer(forecast.arima.sales_1.11, alpha = 0.3, series = "Forecasts") +
  autolayer(validation_1.11, series = "Validation") +
  labs(title = "Dept. 11 ARIMA Model Forecasted Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```




```
# linear model
temp_1.11 <- store_1.11[1:130, 6]
linear_1.11 <- tslm(training_1.11 ~ trend + season + temp_1.11)
summary(linear_1.11)

##
## Call:
## tslm(formula = training_1.11 ~ trend + season + temp_1.11)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3897.2  -669.7   -19.3  1051.8  3971.9
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 20574.852   2717.786   7.570 7.28e-11 ***
## trend          38.956      5.272   7.389 1.61e-10 ***
## season2      -231.512   1766.431  -0.131 0.896072
## season3     -2668.732   1736.989  -1.536 0.128590
## season4     -4628.569   1793.574  -2.581 0.011789 *
## season5     -3113.653   1794.412  -1.735 0.086760 .
## season6     -4309.624   1808.410  -2.383 0.019666 *
## season7     -2275.575   1877.545  -1.212 0.229269
```

```
## season8     -1554.662   1914.180  -0.812 0.419224
## season9     -1286.429   1926.426  -0.668 0.506297
## season10      589.294   2054.530   0.287 0.775025
## season11    -2256.350   2077.984  -1.086 0.280984
## season12    -3128.916   2057.580  -1.521 0.132490
## season13    -2902.844   2076.784  -1.398 0.166252
## season14     6126.514   2136.193   2.868 0.005344 **
## season15      805.606   2258.969   0.357 0.722360
## season16    -3904.021   2152.278  -1.814 0.073640 .
## season17     2317.642   2388.001   0.971 0.334858
## season18    12854.327   2464.035   5.217 1.53e-06 ***
## season19     6994.227   2465.949   2.836 0.005846 **
## season20    10957.494   2567.011   4.269 5.62e-05 ***
## season21    12371.052   2520.426   4.908 5.12e-06 ***
## season22    10842.355   2584.389   4.195 7.31e-05 ***
## season23     6312.552   2538.843   2.486 0.015099 *
## season24     8126.969   2549.973   3.187 0.002086 **
## season25     6226.136   2558.698   2.433 0.017311 *
## season26     5035.189   2580.945   1.951 0.054754 .
## season27     3223.528   2963.033   1.088 0.280071
## season28    -2025.787   2941.067  -0.689 0.493051
## season29    -5996.931   2916.870  -2.056 0.043222 *
## season30    -4357.757   2849.259  -1.529 0.130308
## season31    -5906.005   2769.243  -2.133 0.036177 *
## season32    -6575.308   2512.229  -2.617 0.010689 *
## season33    -7085.924   2638.936  -2.685 0.008897 **
## season34    -8114.638   2545.197  -3.188 0.002079 **
## season35    -7651.499   2458.498  -3.112 0.002616 **
## season36    -4000.025   2194.526  -1.823 0.072278 .
## season37    -6297.231   2267.621  -2.777 0.006905 **
## season38    -5610.229   2197.491  -2.553 0.012684 *
## season39    -5734.587   2230.373  -2.571 0.012090 *
## season40     1724.473   2006.237   0.860 0.392737
## season41    -1931.370   2042.892  -0.945 0.347446
## season42    -2291.093   2005.492  -1.142 0.256870
## season43     4797.366   2095.946   2.289 0.024865 *
## season44     6839.857   1945.526   3.516 0.000743 ***
## season45    13541.862   1948.088   6.951 1.08e-09 ***
## season46     3632.327   1951.429   1.861 0.066559 .
## season47     7416.024   1949.153   3.805 0.000285 ***
## season48    -1362.396   1945.631  -0.700 0.485920
## season49     -609.294   1945.730  -0.313 0.755029
## season50    10217.644   1968.495   5.191 1.69e-06 ***
## season51    -3982.432   1946.989  -2.045 0.044273 *
## season52    -5866.907   1947.202  -3.013 0.003513 **
## temp_1.11      24.555     52.832   0.465 0.643431
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2127 on 76 degrees of freedom
```

```
## Multiple R-squared:  0.9381, Adjusted R-squared:  0.895
## F-statistic: 21.75 on 53 and 76 DF,  p-value: < 2.2e-16

# calculating RMSE
sqrt(mean(linear_1.11$residuals^2))

## [1] 1626.416

# forecasting
temp.new_1.11 <- store_1.11[131:143, 6]
forecast.lm.sales_1.11 <- forecast(linear_1.11, temp.new_1.11, h = 13)

## Warning in forecast.lm(linear_1.11, temp.new_1.11, h = 13): newdata column
names
## not specified, defaulting to first variable required.

forecast.lm.sales_1.11

##          Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## 2012.500       31016.01 27589.42 34442.61 25737.00 36295.03
## 2012.519       25779.63 22348.90 29210.36 20494.24 31065.01
## 2012.538       21842.53 18414.81 25270.25 16561.79 27123.27
## 2012.558       23344.11 19854.97 26833.25 17968.75 28719.48
## 2012.577       21904.31 18471.94 25336.68 16616.39 27192.23
## 2012.596       21359.17 17938.70 24779.64 16089.59 26628.74
## 2012.615       20666.76 17214.75 24118.78 15348.59 25984.94
## 2012.635       19551.78 16068.87 23034.68 14186.01 24917.54
## 2012.654       20206.35 16797.45 23615.26 14954.60 25458.11
## 2012.673       23711.89 20305.24 27118.54 18463.61 28960.17
## 2012.692       21317.12 17860.41 24773.83 15991.71 26642.52
## 2012.712       22165.36 18758.14 25572.57 16916.20 27414.51
## 2012.731       22109.17 18701.78 25516.57 16859.74 27358.60

# plot of forecasted values
autoplot(training_1.11, series = "Training") +
  autolayer(forecast.lm.sales_1.11, alpha = 0.3, series = "Forecasts") +
  autolayer(validation_1.11, series = "Validation") +
  labs(title = "Dept. 11 LM Model Forecasted Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```



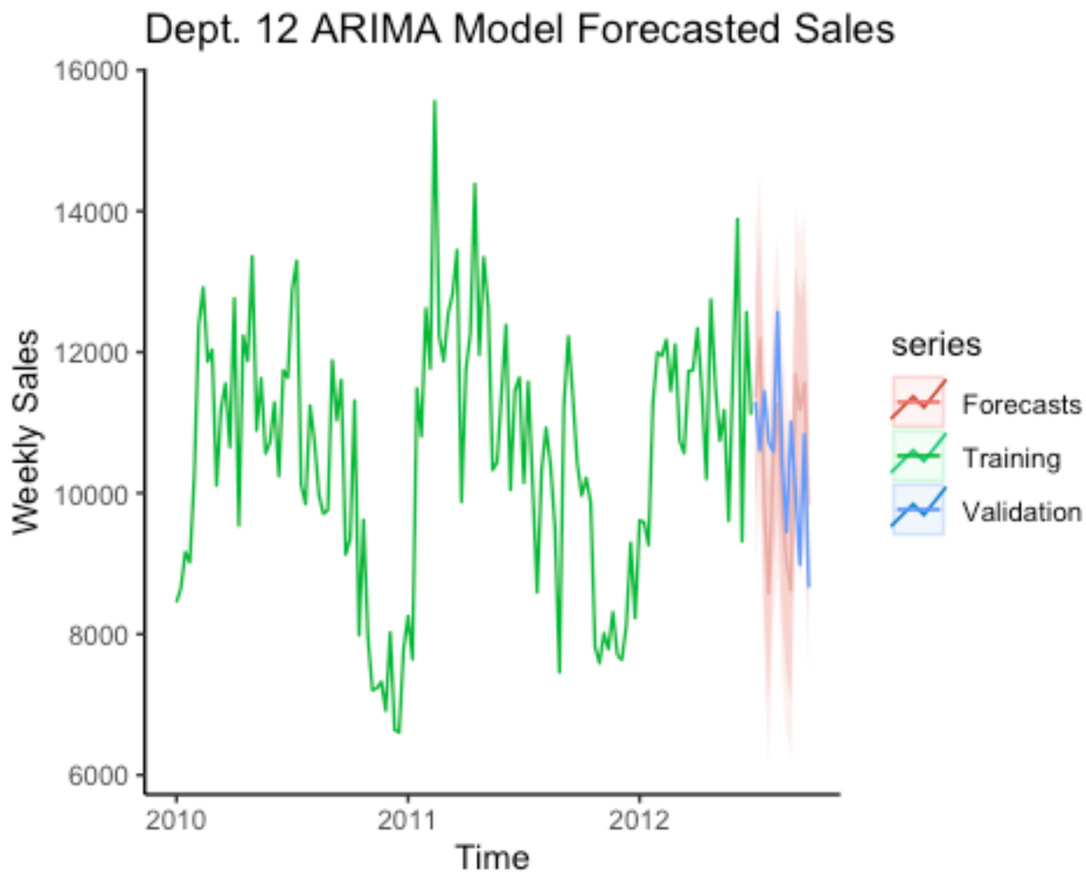
Department 12 Models:

```
# Auto ARIMA model
#predictors.diff_1.1 <- diff(predictors_1.1)
#training.diff_1.1 <- diff(training_1.1)
AutoArima_1.12 <- auto.arima(training_1.12, xreg = predictors_1.12)
summary(AutoArima_1.12)

## Series: training_1.12
## Regression with ARIMA(0,0,0)(0,1,0)[52] errors
##
## Coefficients:
##          xreg
##       71.7328
## s.e.  23.0638
##
## sigma^2 = 1943222:  log likelihood = -674.89
## AIC=1353.78   AICc=1353.94   BIC=1358.49
##
## Training set error measures:
##                      ME    RMSE      MAE       MPE    MAPE      MASE       A
CF1
## Training set -17.67594 1072.84 653.7614 -0.510745 6.12837 0.5566233 0.1362
223
```

```r
# ARIMA model parameters decided by the ACF and PACF plots above
arima_1.12 <- Arima(training_1.12, xreg = predictors_1.12, order = c(0, 1, 5)
, seasonal = c(0, 1, 1))
summary(arima_1.12)
```

```
## Series: training_1.12
## Regression with ARIMA(0,1,5)(0,1,1)[52] errors
##
## Coefficients:
##           ma1      ma2    ma3      ma4      ma5     sma1     xreg
##       -0.9117  -0.0068  0.005  -0.0501  -0.0363  -0.9999  67.6375
## s.e.   0.1355   0.1561  0.172   0.1543   0.1050   0.4438  25.8786
##
## sigma^2 = 936497:  log likelihood = -660.11
## AIC=1336.22   AICc=1338.34   BIC=1354.97
##
## Training set error measures:
##                     ME     RMSE      MAE        MPE     MAPE      MASE
## Training set -50.57084 710.1182 443.4773 -0.7057122 4.156725 0.3775839
##                      ACF1
## Training set -0.009835111
```

```r
# prediction on the arima
new.predictors_1.12 <- as.matrix(store_1.12["Temperature"][131:143,])
forecast.arima.sales_1.12 <- forecast(arima_1.12, xreg = new.predictors_1.12)

# plot of forecasted values
autoplot(training_1.12, series = "Training") +
  autolayer(forecast.arima.sales_1.12, alpha = 0.3, series = "Forecasts") +
  autolayer(validation_1.12, series = "Validation") +
  labs(title = "Dept. 12 ARIMA Model Forecasted Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```




```
# linear model
temp_1.12 <- store_1.12[1:130, 6]
linear_1.12 <- tslm(training_1.12 ~ trend + season + temp_1.12)
summary(linear_1.12)

##
## Call:
## tslm(formula = training_1.12 ~ trend + season + temp_1.12)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1987.66  -464.56   -75.79   443.51  2379.71
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5513.7328  1198.9988   4.599 1.67e-05 ***
## trend          -0.1605     2.3258  -0.069  0.94516
## season2       277.3182   779.2919   0.356  0.72293
## season3      1164.8640   766.3029   1.520  0.13263
## season4      1002.5695   791.2663   1.267  0.20901
## season5      2245.9813   791.6359   2.837  0.00583 **
## season6      2589.4748   797.8113   3.246  0.00174 **
## season7      3832.9632   828.3115   4.627 1.49e-05 ***
```

```
## season8      2004.6691   844.4737   2.374  0.02013 *
## season9      2124.6466   849.8765   2.500  0.01458 *
## season10      904.2687   906.3916   0.998  0.32161
## season11     1237.6258   916.7390   1.350  0.18101
## season12     2005.1012   907.7373   2.209  0.03019 *
## season13      470.3921   916.2095   0.513  0.60915
## season14     1855.2833   942.4189   1.969  0.05264 .
## season15      385.3047   996.5833   0.387  0.70011
## season16     1808.0128   949.5150   1.904  0.06068 .
## season17     1243.9865  1053.5082   1.181  0.24136
## season18     1653.2617  1087.0519   1.521  0.13244
## season19      323.9361  1087.8962   0.298  0.76670
## season20     -241.4492  1132.4816  -0.213  0.83174
## season21    -1001.5520  1111.9299  -0.901  0.37058
## season22       -2.8070  1140.1481  -0.002  0.99804
## season23     1279.9053  1120.0549   1.143  0.25674
## season24    -1393.3360  1124.9649  -1.239  0.21932
## season25      643.1910  1128.8143   0.570  0.57050
## season26      142.6065  1138.6287   0.125  0.90066
## season27     -197.3568  1307.1935  -0.151  0.88039
## season28      760.0332  1297.5030   0.586  0.55977
## season29    -1529.8750  1286.8282  -1.189  0.23819
## season30    -2298.8330  1257.0002  -1.829  0.07135 .
## season31     -591.7979  1221.6998  -0.484  0.62949
## season32      -49.7228  1108.3137  -0.045  0.96433
## season33     -948.0659  1164.2128  -0.814  0.41799
## season34    -1343.1296  1122.8582  -1.196  0.23535
## season35    -2158.7568  1084.6094  -1.990  0.05015 .
## season36     1431.6594   968.1535   1.479  0.14334
## season37     1300.3564  1000.4004   1.300  0.19759
## season38     1350.8967   969.4615   1.393  0.16755
## season39     -459.7473   983.9677  -0.467  0.64167
## season40      203.5106   885.0863   0.230  0.81876
## season41     1139.9345   901.2572   1.265  0.20980
## season42     -521.1550   884.7578  -0.589  0.55758
## season43    -1113.1410   924.6631  -1.204  0.23239
## season44    -1146.2220   858.3028  -1.335  0.18571
## season45    -1027.1378   859.4329  -1.195  0.23575
## season46    -1517.1807   860.9067  -1.762  0.08204 .
## season47    -1166.4817   859.9027  -1.357  0.17895
## season48    -1424.4537   858.3492  -1.660  0.10113
## season49    -1050.6835   858.3927  -1.224  0.22473
## season50    -1040.8167   868.4358  -1.198  0.23445
## season51     -958.8417   858.9480  -1.116  0.26781
## season52     -898.9028   859.0421  -1.046  0.29869
## temp_1.12      69.4466    23.3079   2.980  0.00387 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 938.4 on 76 degrees of freedom
```

```
## Multiple R-squared:  0.8399, Adjusted R-squared:  0.7283
## F-statistic: 7.525 on 53 and 76 DF,  p-value: 2.975e-15
```

```r
# calculating RMSE
sqrt(mean(linear_1.12$residuals^2))
```

```
## [1] 717.5218
```

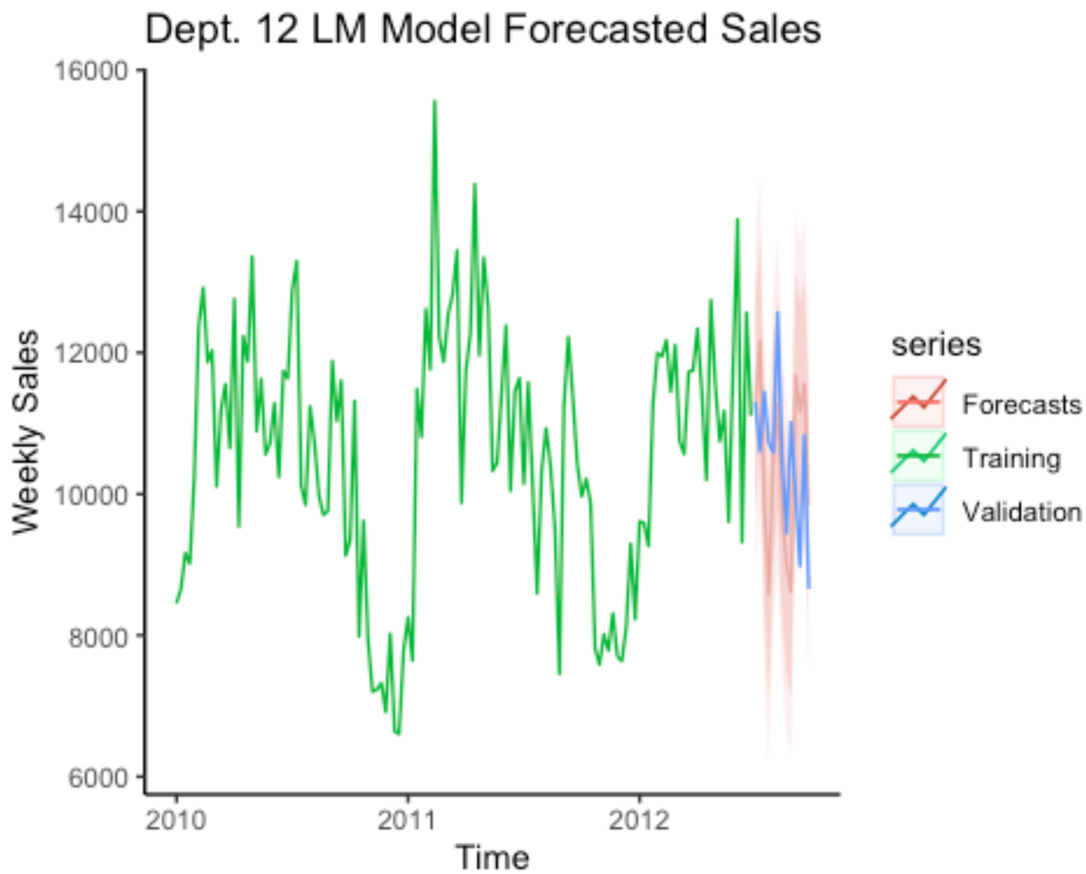
```r
# forecasting
temp.new_1.12 <- store_1.12[131:143, 6]
forecast.lm.sales_1.12 <- forecast(linear_1.12, temp.new_1.12, h = 13)
```

```
## Warning in forecast.lm(linear_1.12, temp.new_1.12, h = 13): newdata column
names
## not specified, defaulting to first variable required.
```

```r
forecast.lm.sales_1.12
```

```
##          Point Forecast     Lo 80    Hi 80    Lo 95    Hi 95
## 2012.500      11275.398  9763.694 12787.10 8946.467 13604.33
## 2012.519      12159.014 10645.487 13672.54 9827.275 14490.75
## 2012.538       9855.056  8342.858 11367.25 7525.364 12184.75
## 2012.558       8586.616  7047.322 10125.91 6215.180 10958.05
## 2012.577      10490.025  8975.773 12004.28 8157.169 12822.88
## 2012.596      11272.919  9763.920 12781.92 8948.155 13597.68
## 2012.615       9750.091  8227.174 11273.01 7403.886 12096.30
## 2012.635       9000.688  7464.143 10537.23 6633.487 11367.89
## 2012.654       8616.164  7112.268 10120.06 6299.261 10933.07
## 2012.673      11683.487 10180.585 13186.39 9368.116 13998.86
## 2012.692      11165.900  9640.913 12690.89 8816.505 13515.30
## 2012.712      11562.124 10058.971 13065.28 9246.367 13877.88
## 2012.731       9833.961  8330.730 11337.19 7518.084 12149.84
```

```r
# plot of forecasted values
autoplot(training_1.12, series = "Training") +
  autolayer(forecast.lm.sales_1.12, alpha = 0.3, series = "Forecasts") +
  autolayer(validation_1.12, series = "Validation") +
  labs(title = "Dept. 12 LM Model Forecasted Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```



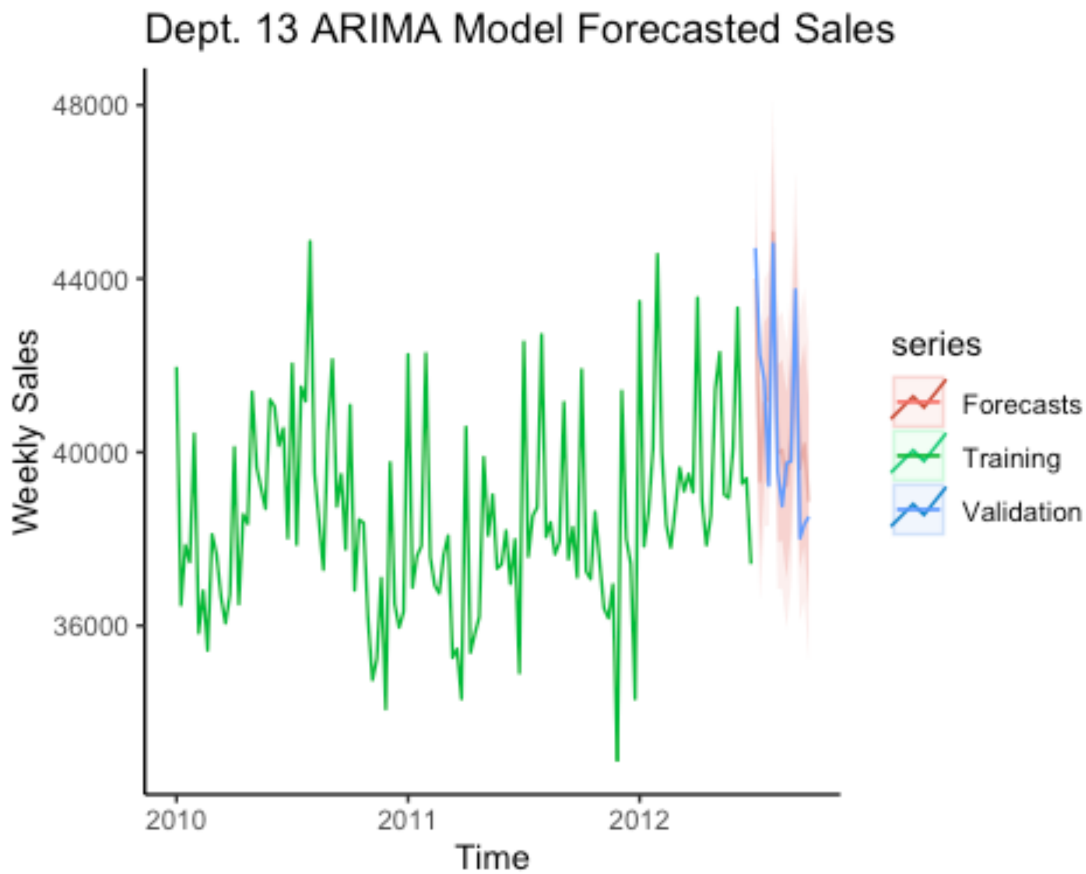
Department 13 Models:

```
# Auto ARIMA model
#predictors.diff_1.1 <- diff(predictors_1.1)
#training.diff_1.1 <- diff(training_1.1)
AutoArima_1.13 <- auto.arima(training_1.13, xreg = predictors_1.13)
summary(AutoArima_1.13)

## Series: training_1.13
## Regression with ARIMA(0,1,1)(0,1,0)[52] errors
##
## Coefficients:
##           ma1      xreg
##       -0.6861  -23.3267
## s.e.   0.0837   22.7037
##
## sigma^2 = 1790171:  log likelihood = -662.86
## AIC=1331.71   AICc=1332.04   BIC=1338.74
##
## Training set error measures:
##                     ME     RMSE      MAE        MPE     MAPE      MASE
## Training set 46.28885 1016.263 629.6389 0.05937672 1.637053 0.3773887
##                     ACF1
## Training set 0.09415731
```

```r
# ARIMA model parameters decided by the ACF and PACF plots above
arima_1.13 <- Arima(training_1.13, xreg = predictors_1.13, order = c(0, 1, 3)
, seasonal = c(0, 1, 1))
summary(arima_1.13)
```

```
## Series: training_1.13
## Regression with ARIMA(0,1,3)(0,1,1)[52] errors
##
## Coefficients:
##           ma1      ma2      ma3     sma1      xreg
##       -0.5709  -0.1185  -0.0213  -0.3703  -17.7384
## s.e.   0.1330   0.1048   0.1264   0.2927   26.2842
##
## sigma^2 = 1619980:  log likelihood = -660.97
## AIC=1333.93   AICc=1335.13   BIC=1348
##
## Training set error measures:
##                    ME     RMSE      MAE         MPE     MAPE      MASE
## Training set 23.52022 947.2167 584.8002 0.001047953 1.517771 0.3505135
##                     ACF1
## Training set -0.01214954
```

```r
# prediction on the arima
new.predictors_1.13 <- as.matrix(store_1.13["Temperature"][131:143,])
forecast.arima.sales_1.13 <- forecast(arima_1.13, xreg = new.predictors_1.13)

# plot of forecasted values
autoplot(training_1.13, series = "Training") +
  autolayer(forecast.arima.sales_1.13, alpha = 0.3, series = "Forecasts") +
  autolayer(validation_1.13, series = "Validation") +
  labs(title = "Dept. 13 ARIMA Model Forecasted Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```




```
# linear model
temp_1.13 <- store_1.13[1:130, 6]
linear_1.13 <- tslm(training_1.13 ~ trend + season + temp_1.13)
summary(linear_1.13)

##
## Call:
## tslm(formula = training_1.13 ~ trend + season + temp_1.13)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2585.58  -647.14    -3.98   651.85  2080.10
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept) 43220.842   1726.882  25.028  < 2e-16 ***
## trend           9.868      3.350   2.946 0.004273 **
## season2     -5664.703   1122.391  -5.047 2.98e-06 ***
## season3     -4545.931   1103.683  -4.119 9.60e-05 ***
## season4     -3920.997   1139.637  -3.441 0.000946 ***
## season5        31.800   1140.170   0.028 0.977823
## season6     -4570.428   1149.064  -3.978 0.000158 ***
## season7     -4931.227   1192.992  -4.133 9.11e-05 ***
```

```
## season8     -5591.046   1216.270  -4.597 1.68e-05 ***
## season9     -4102.502   1224.052  -3.352 0.001254 **
## season10    -3676.859   1305.449  -2.817 0.006182 **
## season11    -5110.465   1320.352  -3.871 0.000228 ***
## season12    -5138.814   1307.387  -3.931 0.000185 ***
## season13    -5436.648   1319.589  -4.120 9.56e-05 ***
## season14     -679.696   1357.338  -0.501 0.617989
## season15    -5087.925   1435.349  -3.545 0.000676 ***
## season16    -4689.307   1367.558  -3.429 0.000982 ***
## season17    -4241.486   1517.336  -2.795 0.006561 **
## season18     -984.480   1565.648  -0.629 0.531365
## season19    -1890.670   1566.864  -1.207 0.231306
## season20    -2777.701   1631.080  -1.703 0.092658 .
## season21    -3567.832   1601.479  -2.228 0.028850 *
## season22    -2279.020   1642.121  -1.388 0.169238
## season23    -1035.127   1613.182  -0.642 0.523019
## season24    -3098.587   1620.253  -1.912 0.059592 .
## season25    -2586.704   1625.798  -1.591 0.115753
## season26    -5110.666   1639.933  -3.116 0.002584 **
## season27      783.087   1882.712   0.416 0.678629
## season28    -3798.715   1868.755  -2.033 0.045571 *
## season29    -1533.103   1853.380  -0.827 0.410717
## season30    -1655.704   1810.420  -0.915 0.363325
## season31     2119.543   1759.578   1.205 0.232105
## season32    -3072.519   1596.271  -1.925 0.057996 .
## season33    -3376.671   1676.781  -2.014 0.047572 *
## season34    -4382.848   1617.219  -2.710 0.008310 **
## season35    -2725.554   1562.131  -1.745 0.085068 .
## season36     -520.809   1394.403  -0.373 0.709816
## season37    -3964.370   1440.847  -2.751 0.007415 **
## season38    -3300.528   1396.287  -2.364 0.020645 *
## season39    -4718.959   1417.179  -3.330 0.001343 **
## season40     -953.068   1274.763  -0.748 0.456982
## season41    -5360.415   1298.054  -4.130 9.24e-05 ***
## season42    -4708.155   1274.290  -3.695 0.000413 ***
## season43    -3843.738   1331.765  -2.886 0.005073 **
## season44    -5844.155   1236.188  -4.728 1.02e-05 ***
## season45    -7215.589   1237.816  -5.829 1.27e-07 ***
## season46    -6955.697   1239.938  -5.610 3.14e-07 ***
## season47    -5661.582   1238.492  -4.571 1.84e-05 ***
## season48    -9311.341   1236.255  -7.532 8.62e-11 ***
## season49    -2156.179   1236.317  -1.744 0.085197 .
## season50    -5671.390   1250.782  -4.534 2.12e-05 ***
## season51    -6055.188   1237.117  -4.895 5.40e-06 ***
## season52    -7446.657   1237.253  -6.019 5.78e-08 ***
## temp_1.13     -25.017     33.570  -0.745 0.458443
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1352 on 76 degrees of freedom
```

```
## Multiple R-squared:  0.7918, Adjusted R-squared:  0.6466
## F-statistic: 5.454 on 53 and 76 DF,  p-value: 1.519e-11
```

```r
# calculating RMSE
sqrt(mean(linear_1.13$residuals^2))
```

```
## [1] 1033.425
```

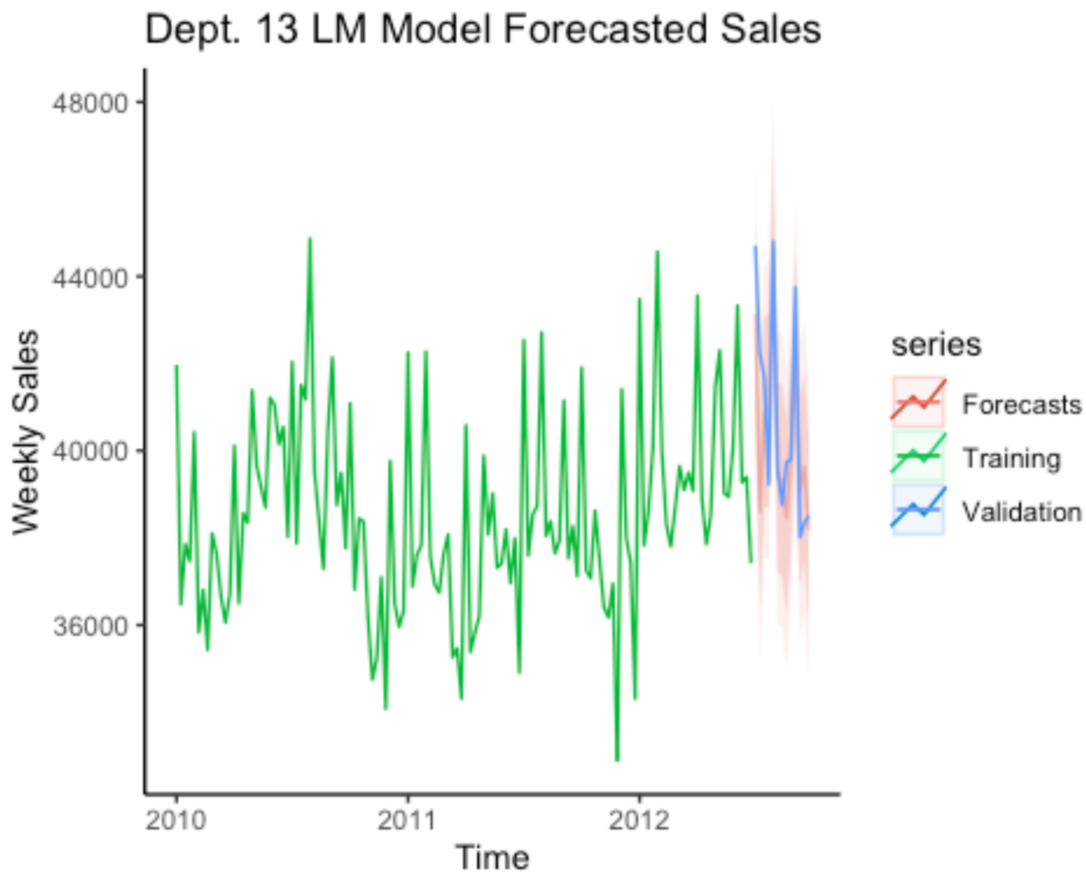
```r
# forecasting
temp.new_1.13 <- store_1.13[131:143, 6]
forecast.lm.sales_1.13 <- forecast(linear_1.13, temp.new_1.13, h = 13)
```

```
## Warning in forecast.lm(linear_1.13, temp.new_1.13, h = 13): newdata column
names
## not specified, defaulting to first variable required.
```

```r
forecast.lm.sales_1.13
```

```
##          Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## 2012.500       43142.50 40965.24 45319.76 39788.21 46496.79
## 2012.519       38597.08 36417.20 40776.97 35238.75 41955.42
## 2012.538       40877.57 38699.59 43055.54 37522.18 44232.95
## 2012.558       40944.70 38727.70 43161.70 37529.19 44360.21
## 2012.577       44659.02 42478.09 46839.95 41299.08 48018.96
## 2012.596       39390.02 37216.65 41563.39 36041.73 42738.31
## 2012.615       39320.63 37127.22 41514.05 35941.47 42699.80
## 2012.635       38451.91 36238.87 40664.95 35042.50 41861.32
## 2012.654       39963.72 37797.70 42129.74 36626.75 43300.69
## 2012.673       42366.71 40202.12 44531.29 39031.95 45701.47
## 2012.692       39072.11 36875.71 41268.50 35688.34 42455.87
## 2012.712       39621.23 37456.29 41786.18 36285.92 42956.55
## 2012.731       38182.90 36017.84 40347.96 34847.41 41518.39
```

```r
# plot of forecasted values
autoplot(training_1.13, series = "Training") +
  autolayer(forecast.lm.sales_1.13, alpha = 0.3, series = "Forecasts") +
  autolayer(validation_1.13, series = "Validation") +
  labs(title = "Dept. 13 LM Model Forecasted Sales",
       x = "Time",
       y = "Weekly Sales") +
    theme_classic()
```



Model Forecasts

```
accuracy(forecast.arima.sales_1.1, validation_1.1)

##                      ME     RMSE       MAE        MPE     MAPE      MASE
## Training set  21.15138 2157.245  954.3406 -0.5351687 4.337541 0.3597749
## Test set     558.06393 2122.165 1251.7916  2.5564757 5.699299 0.4719103
##                    ACF1 Theil's U
## Training set 0.01322383        NA
## Test set     0.02455164  1.243851

accuracy(forecast.arima.sales_1.2, validation_1.2)

##                       ME     RMSE       MAE        MPE     MAPE      MASE
## Training set  -61.27756 1644.405  851.6671 -0.2511672 1.787644 0.4540452
## Test set     -469.15896 1334.388 1119.6226 -1.0674250 2.420985 0.5968991
##                     ACF1 Theil's U
## Training set -0.04005042        NA
## Test set      0.05387770 0.4754589

accuracy(forecast.arima.sales_1.3, validation_1.3)

##                      ME      RMSE       MAE        MPE     MAPE      MASE
## Training set  88.35297  843.5314  458.2682 0.09653082  3.90974 0.4327819
## Test set     448.60214 5262.0309 2887.2858 5.32342144 12.53876 2.7267109
```

```
##                      ACF1 Theil's U
## Training set 0.01085711        NA
## Test set     0.12916644 0.8517941
```

```
accuracy(forecast.arima.sales_1.4, validation_1.4)
```

```
##                        ME     RMSE      MAE        MPE     MAPE      MASE
## Training set   76.57734 1016.042 641.7766  0.1695963 1.738290 0.4332216
## Test set     -440.71691 1205.924 974.7109 -1.2515450 2.595574 0.6579639
##                      ACF1 Theil's U
## Training set -0.0154414        NA
## Test set     -0.2373537 0.4716432
```

```
accuracy(forecast.arima.sales_1.5, validation_1.5)
```

```
##                       ME     RMSE      MAE       MPE     MAPE      MASE
## Training set -30.72939 2549.458 1047.808 0.4313792 4.128701 0.4147436
## Test set     851.81988 2084.068 1450.688 3.8005975 7.062688 0.5742117
##                         ACF1 Theil's U
## Training set  0.0007677673        NA
## Test set     -0.1758945840 0.7742576
```

```
accuracy(forecast.arima.sales_1.6, validation_1.6)
```

```
##                        ME      RMSE       MAE       MPE      MAPE      MASE
## Training set   103.5397  831.0205  429.7156   3.21585  9.327156 0.3586229
## Test set     -1160.9442 1453.4501 1269.2482 -36.75868 39.932352 1.0592623
##                       ACF1 Theil's U
## Training set -0.02217434        NA
## Test set     -0.10321541  2.220785
```

```
accuracy(forecast.arima.sales_1.7, validation_1.7)
```

```
##                       ME     RMSE       MAE       MPE      MAPE      MASE
## Training set -269.0521 2108.843  956.0441 -1.274203  3.859567 0.1924547
## Test set      925.6704 4339.935 3455.9492  3.327876 17.480362 0.6956934
##                       ACF1 Theil's U
## Training set -0.03041530        NA
## Test set     -0.05752251  1.575108
```

```
accuracy(forecast.arima.sales_1.8, validation_1.8)
```

```
##                       ME      RMSE       MAE        MPE     MAPE      MASE
## Training set 100.04547  911.3604  569.2195  0.2388208 1.582977 0.3221513
## Test set     -79.80077 1580.3739 1411.7828 -0.3752218 3.755754 0.7990023
##                       ACF1 Theil's U
## Training set -0.01536931        NA
## Test set     -0.01578996 0.5987785
```

```
accuracy(forecast.arima.sales_1.9, validation_1.9)
```

```
##                       ME      RMSE       MAE       MPE     MAPE      MASE
## Training set  95.69437 1719.250  780.5302 0.4228348 3.092106 0.3050615
## Test set     232.38088 1598.221 1227.9743 0.4060347 3.831337 0.4799400
##                     ACF1 Theil's U
## Training set -0.02348425        NA
## Test set      0.21578220 0.2244506
```

```
accuracy(forecast.arima.sales_1.10, validation_1.10)
```

```
##                        ME     RMSE       MAE        MPE     MAPE     MASE
## Training set   130.8027 1410.495  908.4041  0.2734734 2.898582 0.400411
## Test set     -1928.5263 2672.129 2284.7815 -6.8979737 8.019320 1.007098
##                      ACF1 Theil's U
## Training set -0.006280305        NA
## Test set      0.159955092  1.434442
```

```
accuracy(forecast.arima.sales_1.11, validation_1.11)
```

```
##                         ME     RMSE      MAE        MPE     MAPE      MASE
## Training set    -5.165186 1840.766 1141.668 -0.4183546 4.404359 0.3907944
## Test set     -1572.854944 2401.973 2007.578 -7.9275753 9.831922 0.6871968
##                     ACF1 Theil's U
## Training set  0.02744617        NA
## Test set     -0.39902106 0.6359822
```

```
accuracy(forecast.arima.sales_1.12, validation_1.12)
```

```
##                       ME      RMSE       MAE        MPE      MAPE      MASE
## Training set -50.57084  710.1182  443.4773 -0.7057122  4.156725 0.3775839
## Test set      98.97203 1448.2936 1236.1143  0.1292516 11.919343 1.0524482
##                      ACF1 Theil's U
## Training set -0.009835111        NA
## Test set      0.200639904  1.079634
```

```
accuracy(forecast.arima.sales_1.13, validation_1.13)
```

```
##                        ME      RMSE       MAE          MPE     MAPE      MAS
E
## Training set   23.52022  947.2167  584.8002  0.001047953 1.517771 0.350513
5
## Test set     -224.85128 1314.4269 1055.2115 -0.667298855 2.624314 0.632465
5
##                     ACF1 Theil's U
## Training set -0.01214954        NA
## Test set      0.18182869 0.4253533
```