"Prediction Methods on Walmart Sales Data"

Spencer Tang

5/7/2022

Introduction

The goal of this project is to apply several sets of models to determine if there is a relationship between any of the predictors. I opt to use weekly sales as the predictor and all other variables within the data set as the response. A secondary goal of this project will be to compare several sets of models using RMSE values and determine if other methods may be useful for prediction and see how them compare to using a multiple linear regression model.

Data Description

I found this data set from an author on kaggle, a link to which is provided here: https://www.kaggle.com/datasets/yasserh/walmart-dataset?resource=download

The base data set before further data wrangling has eight columns, Store, Date, Weekly_Sales, Holiday_Flag, Temperature, CPI, Fuel_Price, and Unemployment. This data set takes weekly sales numbers from 45 Walmart stores from the time period of 2/5/2010 to 10/26/2012. CPI stands for consumer price index, a macroeconomic indicator and Fuel_Price is a variable for the cost of a gallon of gas during that week. Temperature is in Fahrenheit and Unemployment is the unemployment rate during that week. The Holiday_Flag is a categorical variable which is set to 1 for the holiday weeks of the Super Bowl, Labor Day, Thanksgiving, and Christmas.

There are 6,435 rows and 8 columns within the data set.

```
dim(dataset)
```

```
## [1] 6435    8
```

On initial investigation, a box plot with weekly sales and the categorical variable of store shows that the store variable may potentially explain a significant amount of the variation in weekly sales. While the top stores may make as much as 2 million dollars in weekly sales, stores on the lower end may make less than half a million dollars.

We can see potential evidence of seasonal trends in the data with large spikes in weekly sales in the Months of November and December.

Because I will use RMSE to compare the performance of our models using cross validation, I have included basic summary statistics for weekly sales to create a reference point for the RMSE values.

```
summary(dataset$Weekly_Sales)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  209986  553350  960746 1046965 1420159 3818686
```

Methods and Results

This data set contains consistent, weekly sales numbers from a period of time from 2010 to 2012 which means I am working with time series data. I first converted the Date column into a Date object and created seasonal dummy variables for both week and month. The `Holiday_Flag` and `Store` variables were both cast as factors along with the seasonal dummy variables.

After cleaning the data, the next step is to split the data into training and test sets; I use all data points from the years of 2010 and 2011 as the training data and the results from 2012 as the test data. It important

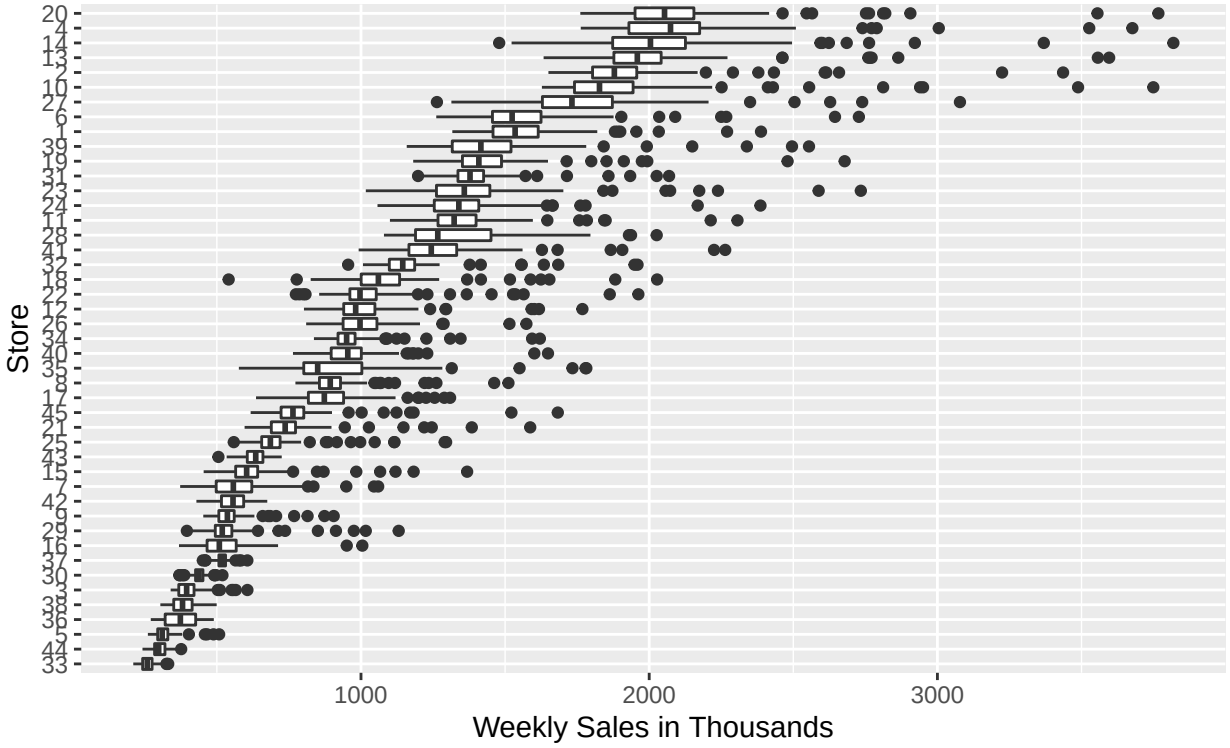

Figure 1: Store vs. Weekly Sales

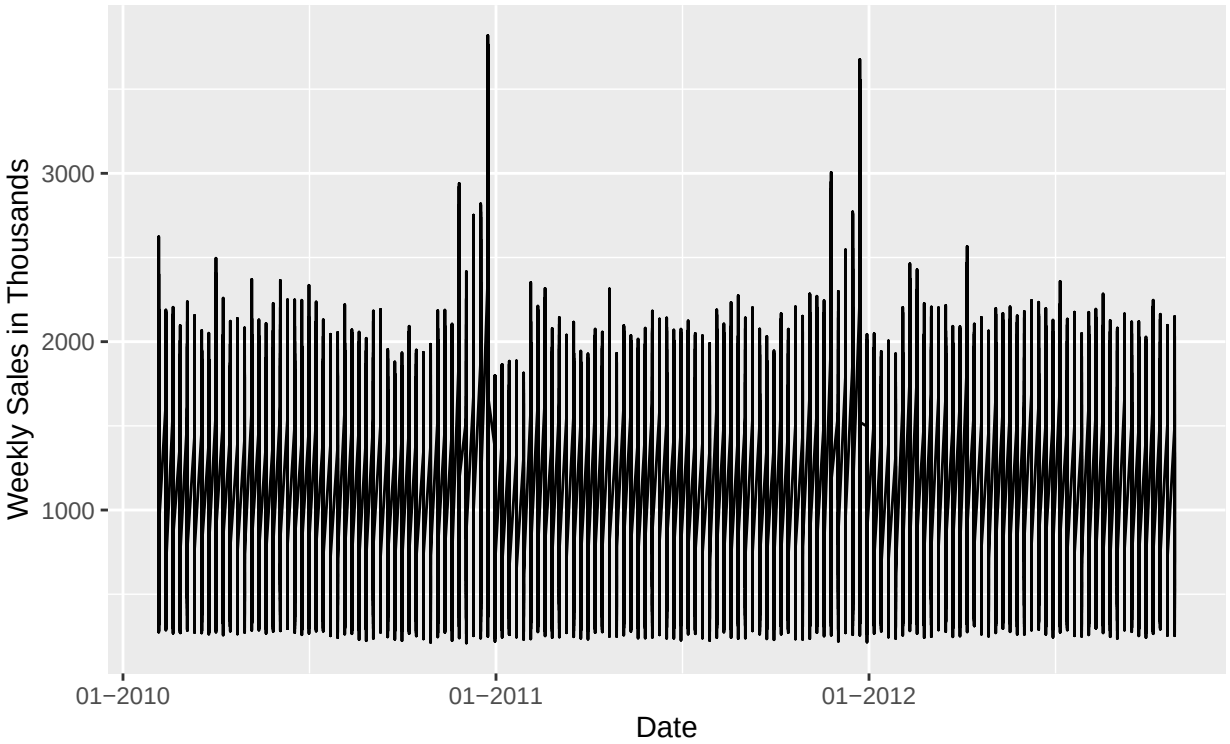

Figure 2: Weekly Sales vs. Date

when working with time series data and cross validation to avoid randomly sampling to preserve the data's inherent aspects of trends and seasonality.

As a first step, I created a full model using `Weekly_Sales` as the response, the other six predictors from the original data set excluding date, and the two seasonal dummy variables.

$$\widehat{WeeklySales} = \beta_0 + \beta_1 HolidayFlag + \beta_2 Temperature + \beta_3 FuelPrice +$$

$$\beta_4 CPI + \beta_5 Unemployment + \beta_6 Store + \beta_7 WeekNumber + \beta_8 Month$$

The initial overall F test was very significant with a p-value near 0 and an adjusted R^2 of 0.9436. Every predictor within this model besides several dummy variables within the categorical predictors had significant p-values less than the chosen alpha of 0.05.

Using the `step()` function and its AIC minimization procedure, the output out of the step function kept all predictors within the model. From here I ran diagnostic checks to evaluate the traditional assumptions of constant variance, linearity, independence, and normality.

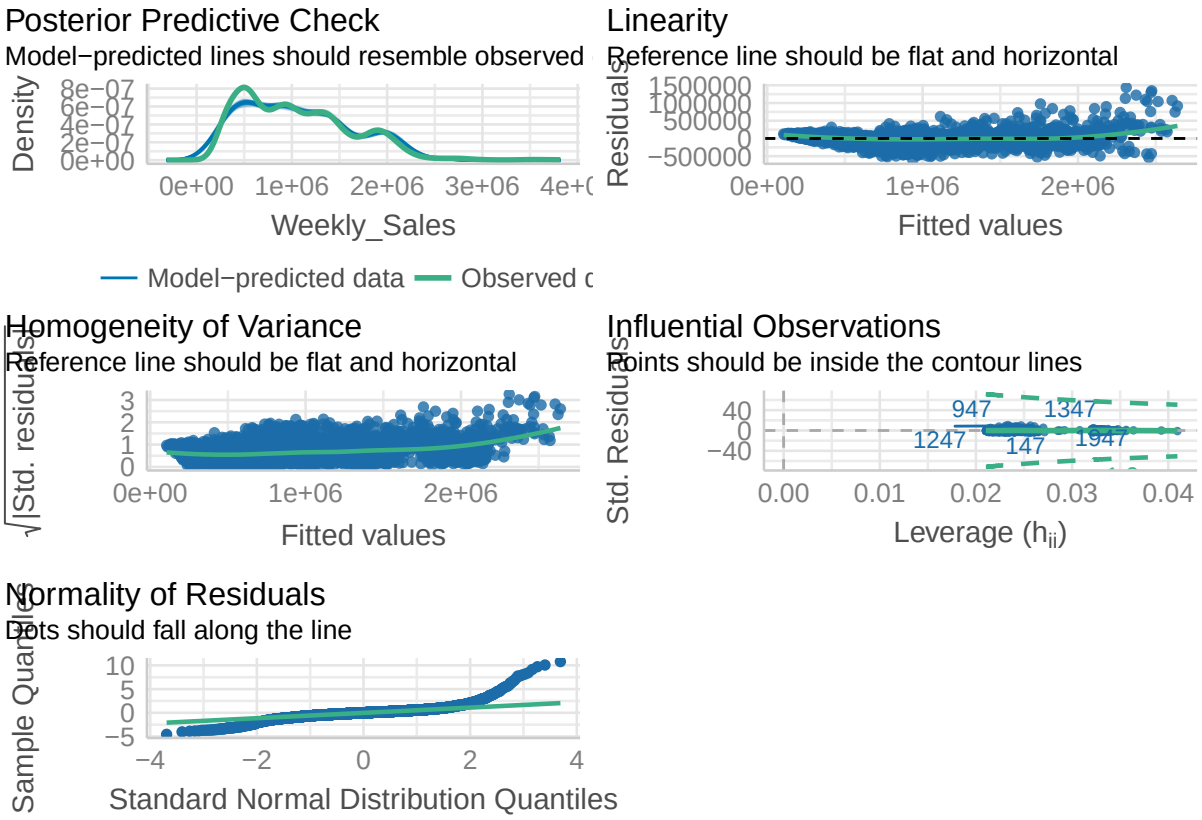

Figure 3: Diagnostics Check of Full Model

We can see from the assorted diagnostic checks that the residuals have a heavy right tail skew and there is apparent heteroscedasticity from the squared standard residuals vs fitted plot. To further reduce the number of predictors within our model we can look at the Variable Inflation Factor(VIF) output .

	Fuel_Price	Temperature	CPI	Unemployment
VIF	5.71	18.3	1723.53	43.18

The `check_collinearity()` function returned high VIF values for all numerical predictors thus I struck them all from the model. I analyzed the adjusted R^2 value from both models to analyze the effects of removing the numerical predictors and saw a small change in adjusted R^2 from 0.9436 to 0.9431.

From this point I ran a Box-Cox method to determine whether a transformation on the response might be necessary.

The Box-Cox plot and associated `powerTransform()` function yielded a estimate for alpha close to 0 so I transformed the response of Weekly_Sales with the `log()` function. After removing the highly correlated predictors and executing a log transformation on the response, I settled on the final model with its output as follows:

```
##
## Call:
## lm(formula = log(Weekly_Sales) ~ Holiday_Flag + Store + Week_Number +
##     Month, data = dataset_train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.66219 -0.04697 -0.00379  0.04643  0.52852
##
## Coefficients: (2 not defined because of singularities)
##                Estimate Std. Error  t value Pr(>|t|)
## (Intercept)   14.164981   0.017553  806.971  < 2e-16 ***
## Holiday_Flag1 -0.025902   0.008445   -3.067 0.002175 **
## Store2         0.226252   0.013877   16.304  < 2e-16 ***
## Store3        -1.362286   0.013877  -98.168  < 2e-16 ***
## Store4         0.289848   0.013877   20.887  < 2e-16 ***
## Store5        -1.596605   0.013877 -115.053  < 2e-16 ***
## Store6         0.016349   0.013877    1.178 0.238818
## Store7        -1.024196   0.013877  -73.805  < 2e-16 ***
## Store8        -0.532700   0.013877  -38.387  < 2e-16 ***
## Store9        -1.055968   0.013877  -76.094  < 2e-16 ***
## Store10        0.219569   0.013877   15.822  < 2e-16 ***
## Store11       -0.129905   0.013877   -9.361  < 2e-16 ***
## Store12       -0.428083   0.013877  -30.848  < 2e-16 ***
## Store13        0.255088   0.013877   18.382  < 2e-16 ***
## Store14        0.312321   0.013877   22.506  < 2e-16 ***
## Store15       -0.885833   0.013877  -63.834  < 2e-16 ***
## Store16       -1.093658   0.013877  -78.810  < 2e-16 ***
## Store17       -0.569043   0.013877  -41.006  < 2e-16 ***
## Store18       -0.345836   0.013877  -24.921  < 2e-16 ***
## Store19       -0.048195   0.013877   -3.473 0.000520 ***
## Store20        0.312592   0.013877   22.526  < 2e-16 ***
## Store21       -0.690995   0.013877  -49.794  < 2e-16 ***
## Store22       -0.396958   0.013877  -28.605  < 2e-16 ***
## Store23       -0.107881   0.013877   -7.774 9.40e-15 ***
## Store24       -0.122009   0.013877   -8.792  < 2e-16 ***
## Store25       -0.775562   0.013877  -55.888  < 2e-16 ***
## Store26       -0.429848   0.013877  -30.975  < 2e-16 ***
```

```
## Store27        0.161596   0.013877   11.645  < 2e-16 ***
## Store28       -0.144958   0.013877  -10.446  < 2e-16 ***
## Store29       -1.043330   0.013877  -75.184  < 2e-16 ***
## Store30       -1.252451   0.013877  -90.253  < 2e-16 ***
## Store31       -0.098008   0.013877   -7.063 1.89e-12 ***
## Store32       -0.282442   0.013877  -20.353  < 2e-16 ***
## Store33       -1.790445   0.013877 -129.022  < 2e-16 ***
## Store34       -0.469661   0.013877  -33.844  < 2e-16 ***
## Store35       -0.491685   0.013877  -35.431  < 2e-16 ***
## Store36       -1.349182   0.013877  -97.224  < 2e-16 ***
## Store37       -1.090700   0.013877  -78.597  < 2e-16 ***
## Store38       -1.438205   0.013877 -103.639  < 2e-16 ***
## Store39       -0.091909   0.013877   -6.623 3.94e-11 ***
## Store40       -0.469148   0.013877  -33.807  < 2e-16 ***
## Store41       -0.220421   0.013877  -15.884  < 2e-16 ***
## Store42       -1.036527   0.013877  -74.693  < 2e-16 ***
## Store43       -0.883098   0.013877  -63.637  < 2e-16 ***
## Store44       -1.664007   0.013877 -119.910  < 2e-16 ***
## Store45       -0.668187   0.013877  -48.150  < 2e-16 ***
## Week_Number2  -0.050827   0.020687   -2.457 0.014049 *
## Week_Number3  -0.056186   0.020687   -2.716 0.006632 **
## Week_Number4  -0.079549   0.020687   -3.845 0.000122 ***
## Week_Number5   0.281467   0.058511    4.811 1.56e-06 ***
## Week_Number6   0.337305   0.057744    5.841 5.55e-09 ***
## Week_Number7   0.340501   0.057744    5.897 3.99e-09 ***
## Week_Number8   0.277894   0.057589    4.825 1.44e-06 ***
## Week_Number9   0.235653   0.054732    4.306 1.70e-05 ***
## Week_Number10  0.210876   0.053746    3.924 8.86e-05 ***
## Week_Number11  0.202268   0.053746    3.763 0.000170 ***
## Week_Number12  0.172081   0.053746    3.202 0.001376 **
## Week_Number13  0.176436   0.050672    3.482 0.000503 ***
## Week_Number14  0.269825   0.049605    5.439 5.63e-08 ***
## Week_Number15  0.233610   0.049605    4.709 2.56e-06 ***
## Week_Number16  0.241648   0.049605    4.871 1.15e-06 ***
## Week_Number17  0.184056   0.049605    3.710 0.000209 ***
## Week_Number18  0.177122   0.046257    3.829 0.000130 ***
## Week_Number19  0.179141   0.045086    3.973 7.20e-05 ***
## Week_Number20  0.132606   0.045086    2.941 0.003287 **
## Week_Number21  0.141396   0.045086    3.136 0.001723 **
## Week_Number22  0.193129   0.041373    4.668 3.13e-06 ***
## Week_Number23  0.197464   0.040060    4.929 8.56e-07 ***
## Week_Number24  0.172643   0.040060    4.310 1.67e-05 ***
## Week_Number25  0.155002   0.040060    3.869 0.000111 ***
## Week_Number26  0.149587   0.035830    4.175 3.04e-05 ***
## Week_Number27  0.165742   0.034305    4.831 1.40e-06 ***
## Week_Number28  0.133734   0.034305    3.898 9.83e-05 ***
## Week_Number29  0.114107   0.034305    3.326 0.000887 ***
## Week_Number30  0.079734   0.034305    2.324 0.020157 *
## Week_Number31  0.087140   0.029255    2.979 0.002912 **
## Week_Number32  0.073566   0.027366    2.688 0.007211 **
## Week_Number33  0.057575   0.027366    2.104 0.035444 *
## Week_Number34  0.066443   0.027366    2.428 0.015225 *
## Week_Number35  0.044081   0.020687    2.131 0.033154 *
## Week_Number36  0.093356   0.018406    5.072 4.10e-07 ***
```

```
## Week_Number37  0.047904   0.018406    2.603 0.009283 **
## Week_Number38 -0.004034   0.017915   -0.225 0.821862
## Week_Number39 -0.030368   0.017915   -1.695 0.090124 .
## Week_Number40 -0.149515   0.035329   -4.232 2.36e-05 ***
## Week_Number41 -0.147009   0.035329   -4.161 3.23e-05 ***
## Week_Number42 -0.154662   0.035329   -4.378 1.23e-05 ***
## Week_Number43 -0.154629   0.035329   -4.377 1.23e-05 ***
## Week_Number44 -0.174896   0.030450   -5.744 9.89e-09 ***
## Week_Number45 -0.205432   0.028639   -7.173 8.57e-13 ***
## Week_Number46 -0.223639   0.028639   -7.809 7.17e-15 ***
## Week_Number47 -0.057806   0.027690   -2.088 0.036889 *
## Week_Number48  0.122981   0.020687    5.945 2.98e-09 ***
## Week_Number49  0.175347   0.017915    9.788  < 2e-16 ***
## Week_Number50  0.261043   0.017915   14.571  < 2e-16 ***
## Week_Number51  0.417380   0.017915   23.297  < 2e-16 ***
## Week_Number52  0.321430   0.018406   17.463  < 2e-16 ***
## Week_Number53 -0.038331   0.022344   -1.715 0.086326 .
## Month2        -0.219792   0.054732   -4.016 6.02e-05 ***
## Month3        -0.156381   0.050672   -3.086 0.002040 **
## Month4        -0.170169   0.046257   -3.679 0.000237 ***
## Month5        -0.100292   0.041373   -2.424 0.015388 *
## Month6        -0.078351   0.035830   -2.187 0.028815 *
## Month7        -0.057090   0.029255   -1.951 0.051069 .
## Month8         0.012733   0.020687    0.616 0.538244
## Month9               NA         NA       NA       NA
## Month10        0.181883   0.030450    5.973 2.51e-09 ***
## Month11        0.285746   0.022344   12.788  < 2e-16 ***
## Month12              NA         NA       NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.09813 on 4393 degrees of freedom
## Multiple R-squared:  0.9734, Adjusted R-squared:  0.9728
## F-statistic:  1519 on 106 and 4393 DF,  p-value: < 2.2e-16
```

The final model leaves only categorical variables behind and thus its interpretation may not be very straightforward or even useful in some situations. The Holiday_Flag1 predictor is negative, which says that weeks which fall on four designated holidays make fewer weekly sales than weeks which do not fall on those four designated holidays. This conclusion is counter intuitive to summary statistics of the data set which state that Weekly Sales during holiday weeks are on average greater than that of non holiday weeks. I suspect that there may be additional confounding factors within the model which have yielded these coefficient estimates.

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  215359  575866 1018538 1122888 1555213 3004702
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  209986  551378  956211 1041256 1414344 3818686
```

Each Store, Week_Number, and Month's coefficient estimates can be interpreted relative to to the first baseline or reference Store, Week_Number, and Month on the condition that the other predictors are held fixed. For example, the month of May has a negative sign next to its estimate, thus we can conclude that Weekly Sales in May are less than the Weekly Sales in January, with all other predictors held fixed. Two of the dummy variables within the Month factor return NA values because the months of September and December show perfect correlation with another predictor within the model.

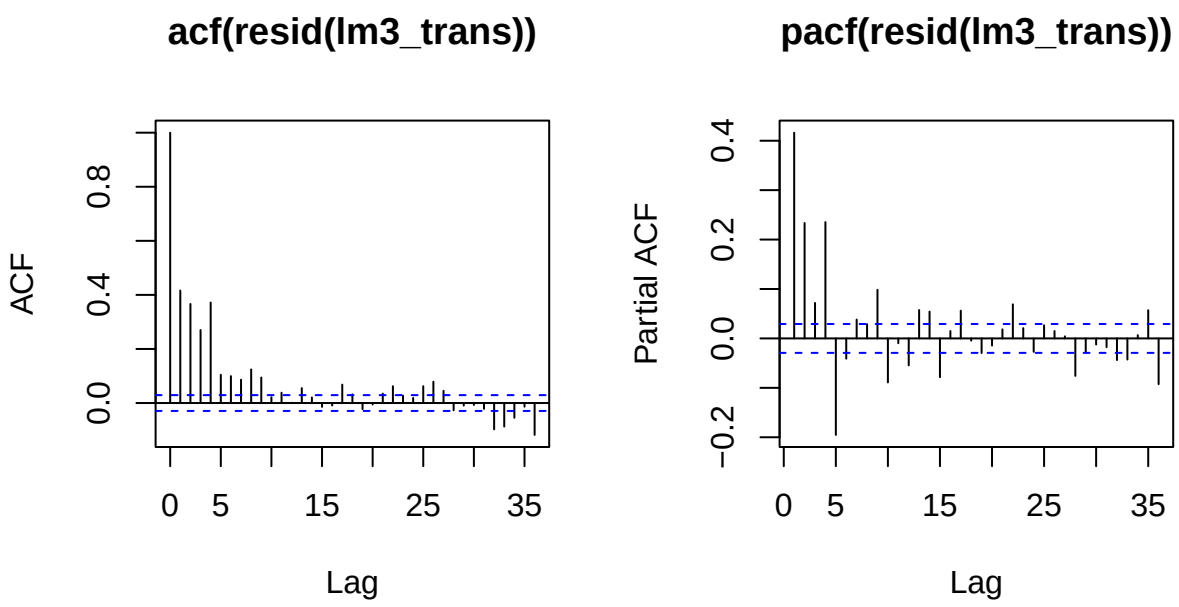

Figure 4: acf and pacf plots

Taking both ACF and PACF plots of the residuals of our final model show that our traditional assumptions of independence for linear regression modeling are not satisfied, with clear evidence of autocorrelation to previous lagged values.

The final diagnostics check of the model show that the transformation and pruning of the full model down to the final model has improved on the full model's heteroscedasticity, but the residuals and variance of the data is not completely random, displaying heavy clustering in the residuals vs fitted values plot. Normality assumptions have also somewhat improved but are still far from ideal.

Random Forest Approach

In this subsection I explore the usefulness of a Random Forest method to determine whether its use could possibly improve predictive capabilities and provide us information on which predictors in the model have the most influential effect on the response.

The initial random forest model, which uses the same response and predictors as the linear regression full model, resulted in an output with negative % Var explained on the out of bag data, a sign that the model is over fitting. After applying a `vif()` graphical analysis, I removed the least influential predictor and reran the model. The MSE of the random forest model has stabilized by 500 bootstrapped trees, and the mtry = 2 default number of subsetted predictors was deemed to be the ideal parameter value. Further removing less influential predictors beyond `Week_Number` decreased OOB performance.

```
##
## Call:
##  randomForest(formula = Weekly_Sales ~ . - Date - Week_Number,      data = dataset_train, ntree = 50
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
```

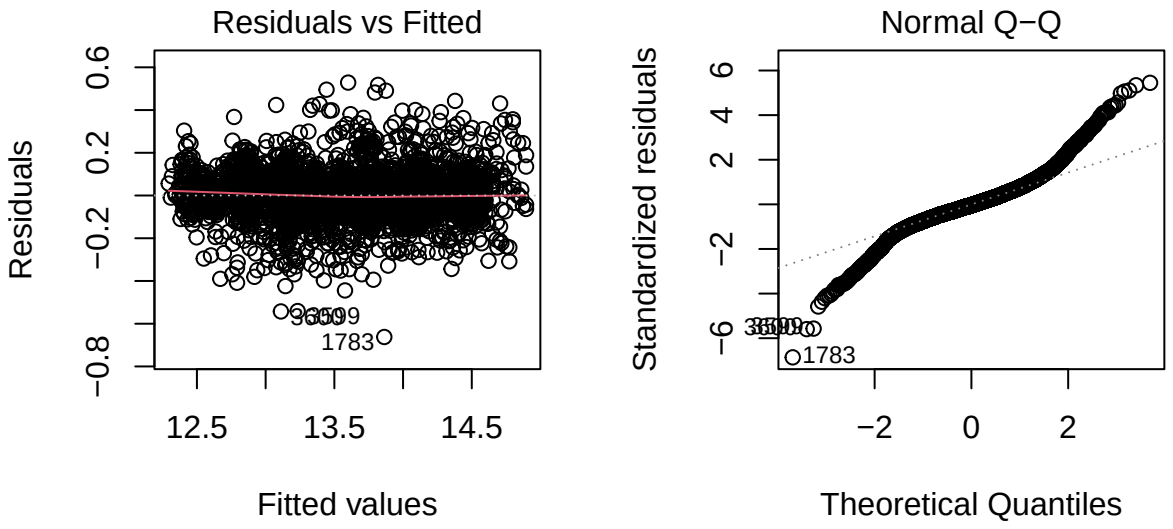

Figure 5: Final Model Diagnostics

```
##           Mean of squared residuals: 191979818431
##                      % Var explained: 42.09
```

From the VIP plot of the predictors we can see that Store is by far the most important predictor. Unlike the linear regression model, the random forest model retained numerical predictors such as CPI and Unemployment. However, the relatively poor built in crossvalidation on the out of bag data can be explained by the process the random forest takes while creating the bootstrapped decision trees. While the data is clearly seasonal, the process of randomly sampling observations in the random forest process does not preserve the elements of autocorrelation nor does it keep the seasonal effects of the sharp increases in weekly sales towards the end of the year.

Final Prediction and Model Analysis

Here we lay out the results of the cross-validation on the withheld 2012 sales data. The removal of the highly correlated predictors within the model makes very minor increases in the cross validation RMSE results while the log transformation actually decreases our RMSE value. The high RMSE of the Random Forest model shows its inadequacy in dealing with time series data without further changes in either how the model incorporates its predictors or samples observations. In real world terms, if we make a prediction using the best model, the transformed and reduced linear regression (final) model, we would be off on average by $97,726.79 from the actual value. Considering that the average weekly sales for the data was found to be $1,046,965, this model's predictions are not too far off from the actual values.

	Full Model	Reduced Model	Full Reduced and Transformed Model	Random Forest Model
RMSE Values	104821.7	104980.4	97726.79	433795.9

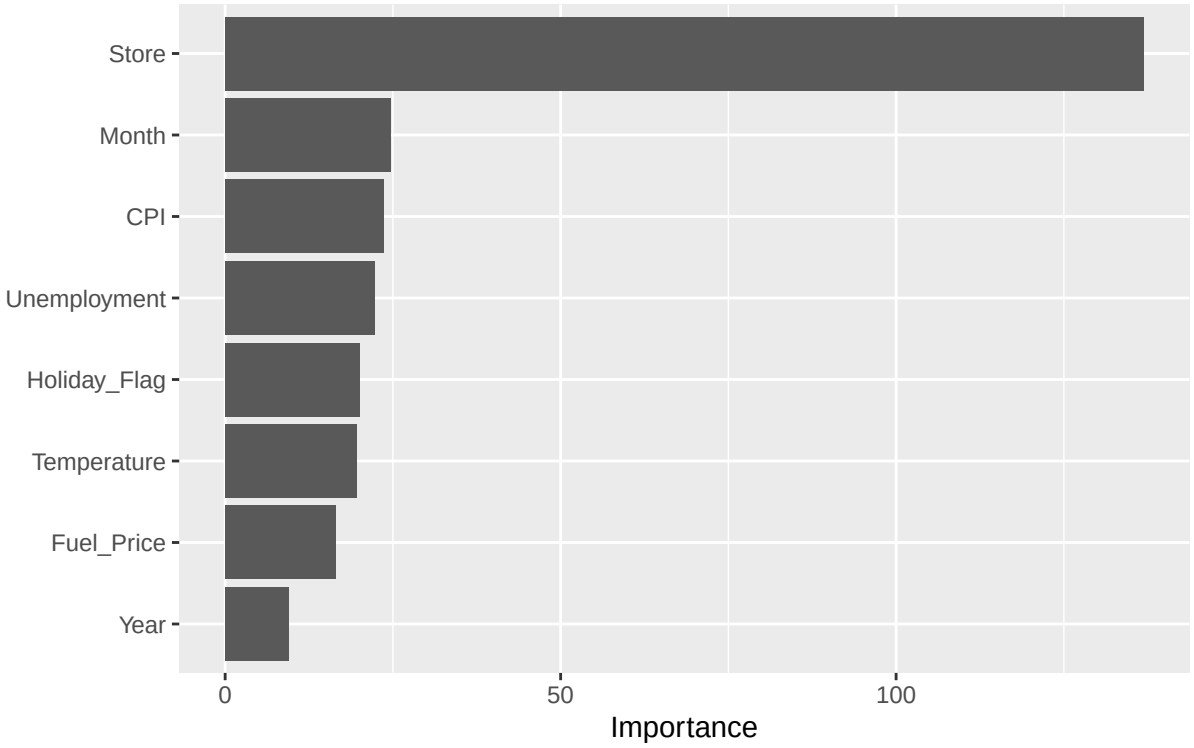

Figure 6: Variable Importance Plot

Conclusion

We have determined that a linear regression model with Weekly Sales as the response and Holiday Flag, Month, Week,(all seasonal dummy variables) and Store as predictors explain a high percentage of the variance in the response. Unfortunately the traditional assumptions of linear regression do not hold up well in regards to non-stationary time series data, thus while we can make interpretations with the coefficients and sign values, it is difficult to determine relationships between each predictor and the response in confidence. However, with good predictive performance with cross-validation and a high R^2 value, the model has proven itself as an effective model for the purposes of prediction and outperformed the Random Forest model.

Addendum: ARIMA Model Implementation

An ARIMA model is one useful option for uni-variate prediction of a response variable. This model uses a linear combination of past(lagged) variable values as well as a component with the weighted moving average of the past forecast errors. The seasonal ARIMA model I will use here also includes non-seasonal and seasonal terms. Our objective in this section is to predict the sum of all total weekly sales of all 45 Walmart stores over time.

I conducted a unit root test to determine both if the data set can be considered stationary as well to determine whether a first difference may be necessary. In a Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test, the null hypothesis is that the data is stationary and the alternative hypothesis says that the data is not stationary. Furthermore, the obvious seasonal effect of sharp sales increases towards the end of year very likely necessitates the use of one seasonal difference, which we will apply as gap of one year.

The variance of the time series data does not seem to be drastically increasing or decreasing over time so a transformation may not be necessary here.

```
## # A tibble: 1 x 2
##   kpss_stat kpss_pvalue
##       <dbl>       <dbl>
## 1    0.0498         0.1
```

With a p-value above the chosen alpha of 0.05, we fail to reject the null hypothesis that the data is stationary and no first order difference is necessary. I now apply the `ARIMA()` function, which will automatically produce the coefficients of an ARIMA(p,d,q)(P,D,Q) model.

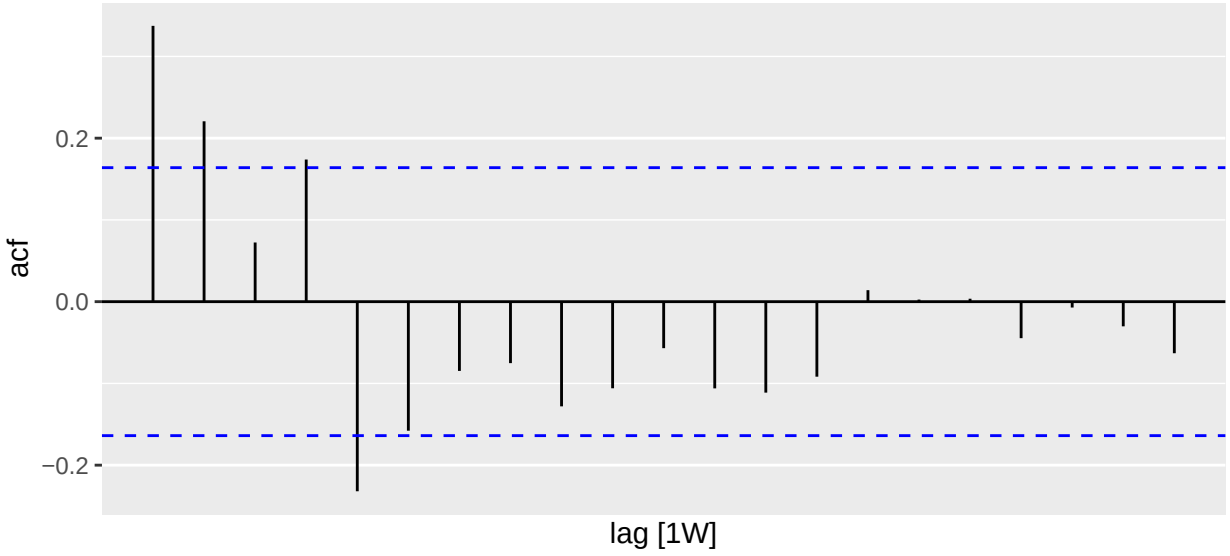

Figure 7: ACF Plot of the Time Series Data

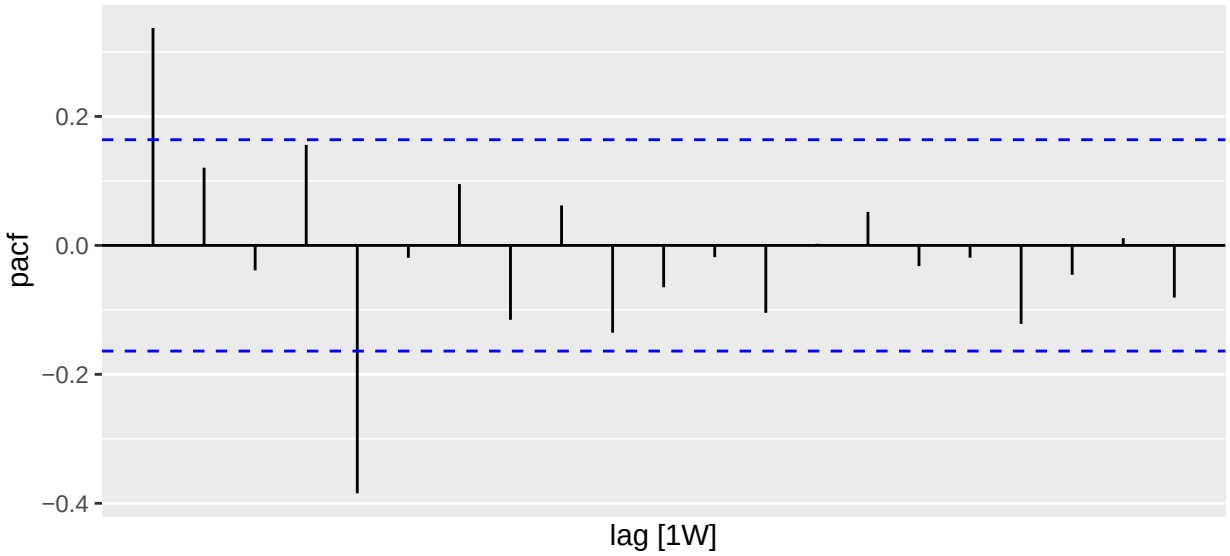

Figure 8: PACF Plot of the Time Series Data

```
## Series: Total_Sales_Per_Week
## Model: ARIMA(0,1,1)(0,1,0)[52]
##
## Coefficients:
##           ma1
##       -0.8923
## s.e.   0.0439
##
## sigma^2 estimated as 3.441e+12:  log likelihood=-1426.98
## AIC=2857.95   AICc=2858.09   BIC=2862.95
```

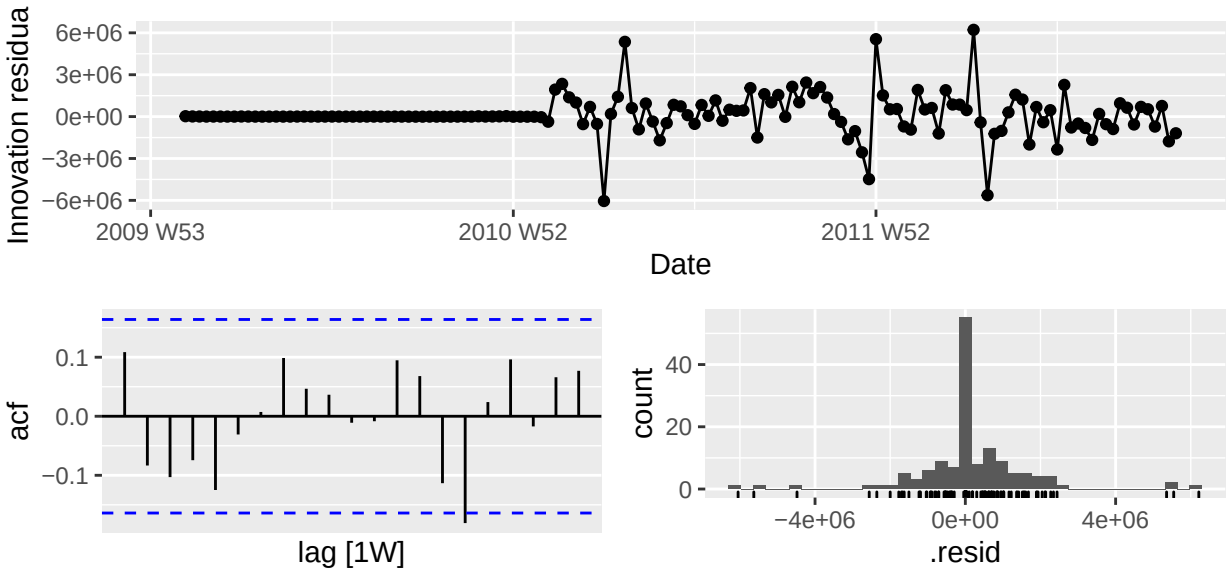

Figure 9: ARIMA Model Diagnostics

A check of the residuals inside the ARIMA(0,1,1)(0,1,0)_(52) model show that almost all acf values of the residuals are within the threshold of the ACF plot and the residuals follow an approximately normal distribution. This model uses one seasonal difference, one first difference, and one moving average term. While the KPSS test concluded that the data was stationary, the model selection procedure in the `ARIMA()` function still found that an ARIMA model with a first order difference was the best model.

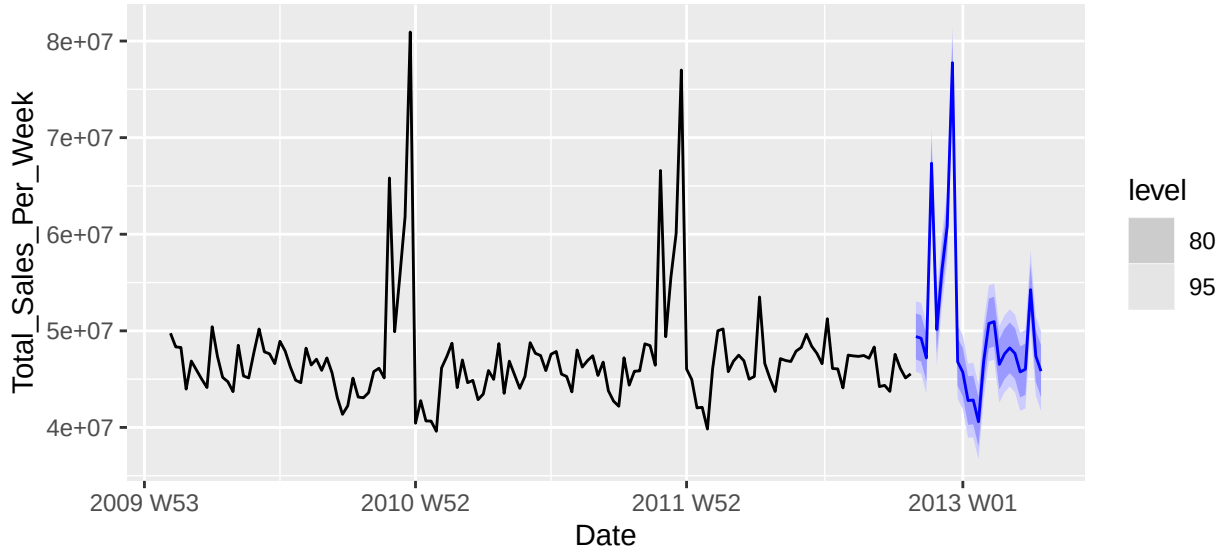

Figure 10: Forecasted Weekly Sales Values

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## 39599853 44880588 46243900 47113419 47792025 80931416
```

```
## # A tibble: 1 x 10
##   .model    .type         ME     RMSE     MAE   MPE  MAPE  MASE RMSSE  ACF1
##   <chr>     <chr>      <dbl>    <dbl>   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 autoarima Training 149088. 1463328. 831107. 0.329  1.74 0.580 0.732 0.109
```

It is clear from the RMSE metrics and plot of estimated future values that the ARIMA model serves as a good predictor of future total weekly sales across all Walmart stores. A good way to further investigate the efficacy of this model would be to split the data into training and test sets, applying the earlier cross-validation procedures used for the first two models to verify predictive efficacy and analyze whether the RMSE values significantly decrease.