STAT 4214 Final Individual Project

Matthew Grace

12/11/2020

Electricity Demand

Summary of Problem

The data presented for this problem is 364 observations representing 364 days of a calendar year, starting on Sunday, January 1. Each observation contains a reading of the rate of electricity delivered to customers of the company Gulf Energy in a particular area of Alabama; Gulf Energy is the provider of the data. These readings for electricity rate are measure in megawatts, and naturally are represented with the column name *MW*. The other column containing data for each observation is the average daily temperature of the customers for which this electricity is delivered on the particular day, and has column name *temp*, which is a temperature measure in degrees Fahrenheit.

The main task with this data will be to forecast electricity demand (MW) for this particular area of Alabama. There will be a focus on the peak electricity demand. What are the necessary predictors in order to accurately forecast the electricity demand? Additionally, can the electricity demand be accurately forecasted? If so, what is the prediction interval of this demand?

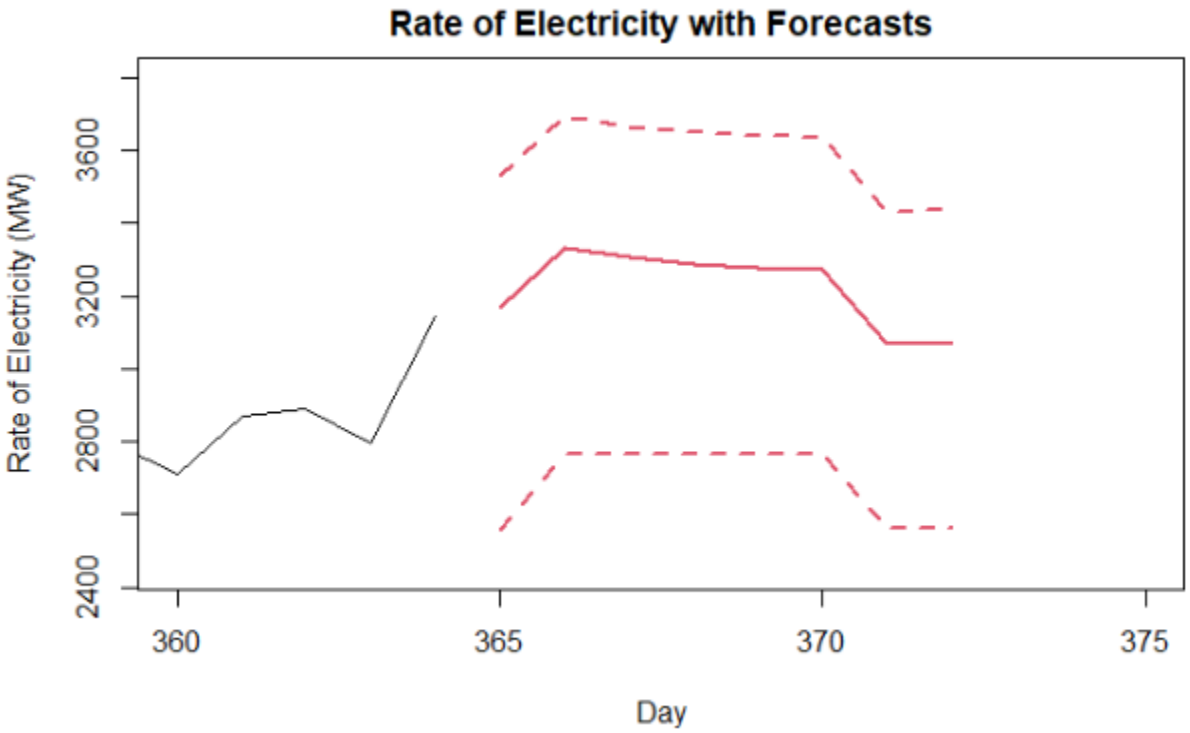
In order to answer these questions, I came to a 2 model solution. There was a clear linear regression that could be used to model electricity demand as a function of temperature, however, for different temperature ranges. A multiple regression with interactions that were dummy variables for temperature ranges very accurately captured the electricity demand. In order to forecast future electricity demand, a forecast of temperature was needed. This temperature forecast was done with an autoregressive time series model. Once forecasted temperatures were computed along with their 95% prediction intervals, days 365-372 had their electricity demand forecasted.

Due to there being forecasting for the temperature variable, the the 95% prediction intervals for the multiple regression model used to forecast electricity had additive prediction intervals. Let me explain. The forecasted temperatures had a lower and upper bound for their 95% prediction intervals, as well as an actual forecasted temperature. In order to get a correct prediction interval on the forecasted rate of electricity, I had to forecast electricities using the lower bound, upper bound, and predicted temperatures for each day in the forecast. While this sounds confusing, the analysis will visually be able to give a better representation of this scenario.

Ultimately, the following table shows the forecasted rate of electricity with upper and lower bounds on the 95% prediction interval.

Day <int>	Forecasted_Electric_Rate <dbl>	Prediction_Interval_Upper_Bound <dbl>	Prediction_Interval_Lower_Bound <dbl>
365	2924.069	3531.414	2560.059
366	3128.482	3693.640	2765.734
367	3128.572	3666.134	2765.989
368	3128.629	3648.744	2766.145
369	3128.661	3638.982	2766.230
370	3128.675	3634.992	2766.264
371	2924.394	3432.207	2560.995
372	2924.382	3435.964	2560.962

These values are plotted below.



Data Analysis

EDA Phase:

In this phase of analysis, visualization and feature creation will be done to aid the future analysis of the data relating to the end-goal of accurate forecasting.

To begin, the data will be loaded and the head of the data will be viewed.

```
elec <- read.csv('elec.csv')
head(elec)
```

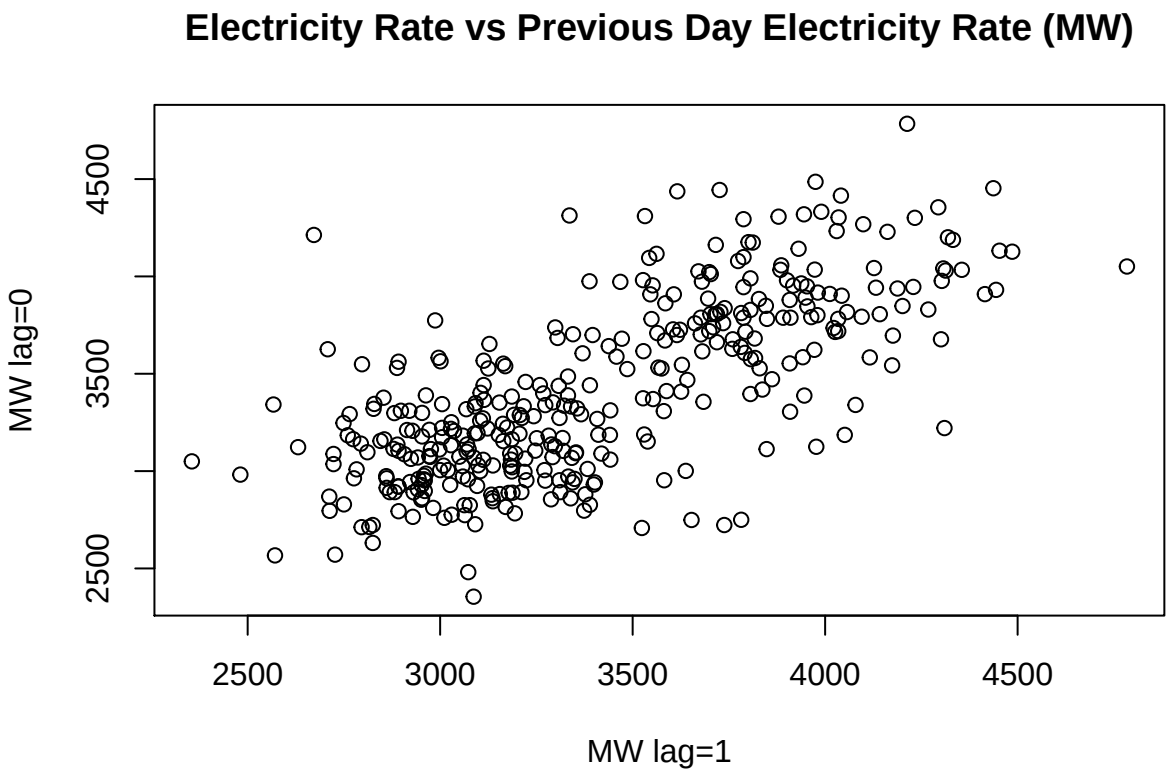
```
##      MW temp
## 1 2672   67
## 2 4213   44
## 3 4784   39
## 4 4051   53
## 5 3186   61
## 6 3383   64
```

The data begins on a Sunday. Time is a critical component for this problem, and it is likely that accurate forecasting will require more than 1 independent variable (*temp*). Looking at the context of this data, customers have differing electric usage depending on when they are present or absent from their households or place of business. The days of the week are a societal strucutre that could potentially be an indicator for presence or absence of customers from their source of electricity. A column for this feature will be added.

```
# DOW is the day of the week
elec$DOW <- rep(c('Sun', 'Mon', 'Tue', 'Wed', 'Thur', 'Fri', 'Sat'), 52)
```

Visualization Since time is a critical component of this model, I will start by plotting the MW values againts the previous day MW values.

```
# time variable to create a lag for the MW values
time.l1 <- 2:nrow(elec)
plot(elec$MW[time.l1-1], elec$MW[time.l1], xlab = 'MW lag=1', ylab = 'MW lag=0',
     main='Electricity Rate vs Previous Day Electricity Rate (MW)')
```

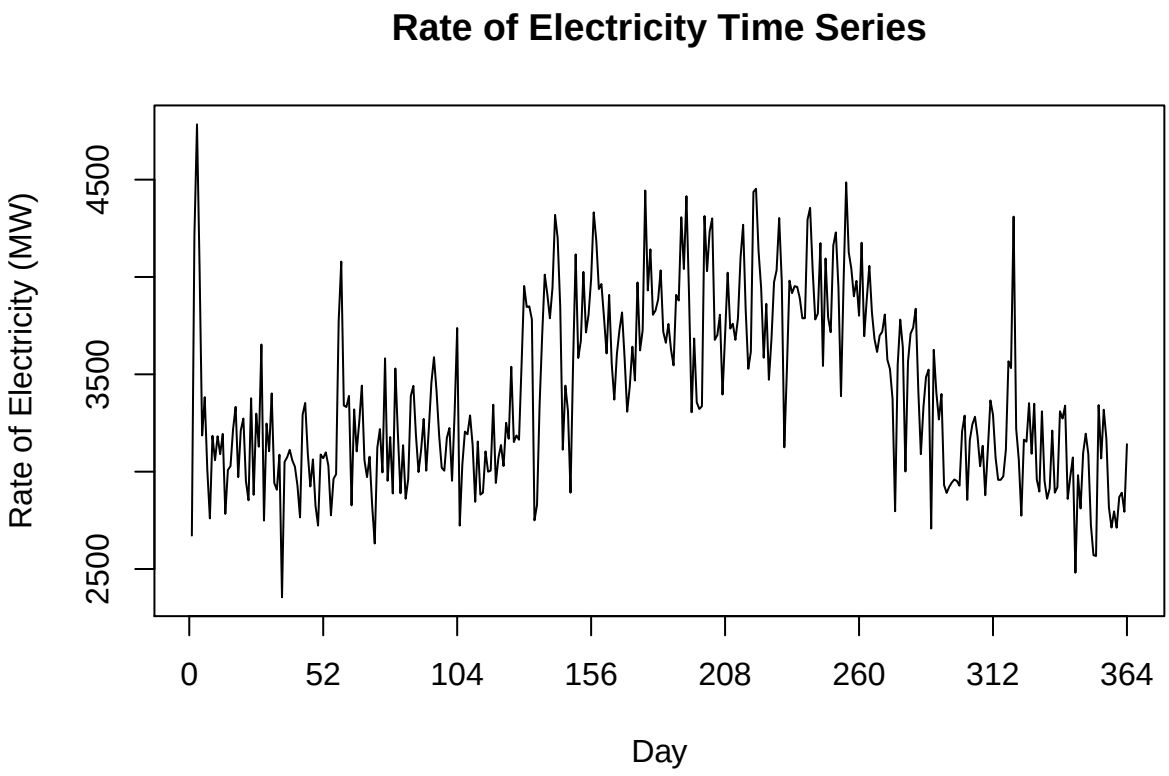



```
cor(elec$MW[time.l1], elec$MW[time.l1-1])
```

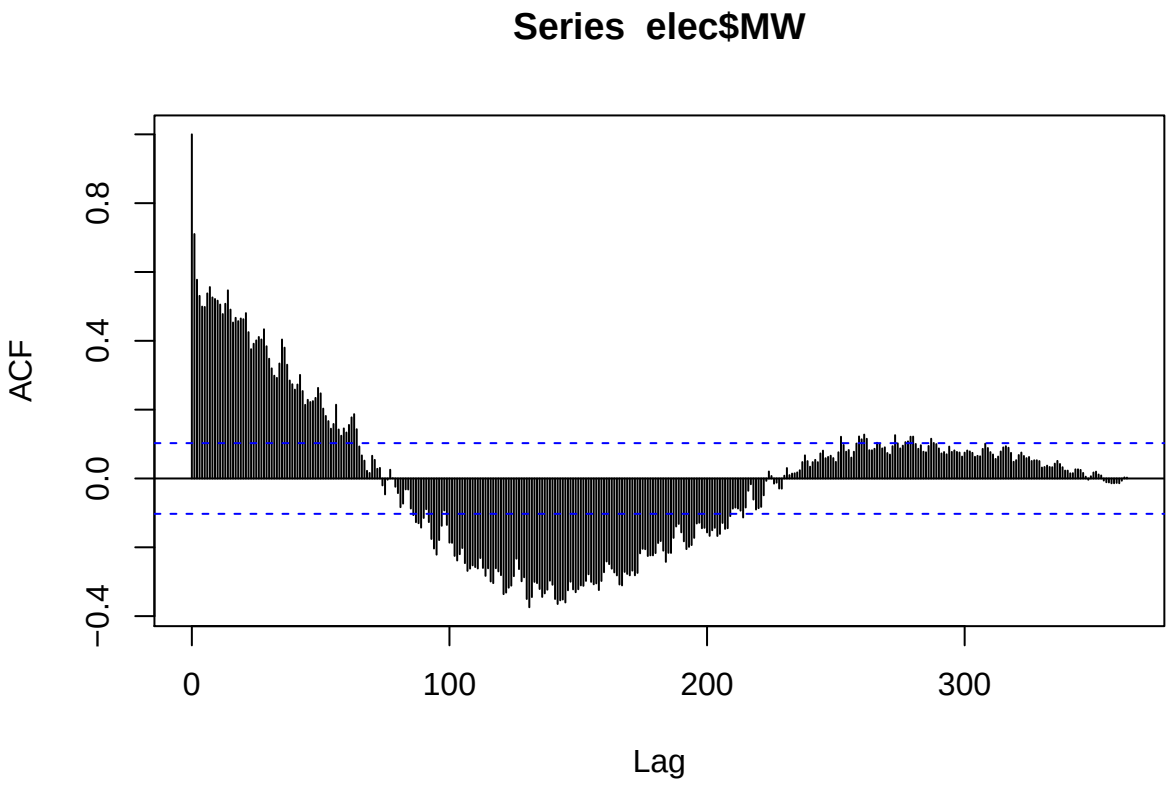
```
## [1] 0.713219
```

At a minimum, a lag=1 autoregressive term seems to have a suitable correlation strength as shown by the plot and the computed correlation of 0.71. I want to see the viability of a time series for this problem, and also see if classical regression for MW vs *temp* is a viable model choice.

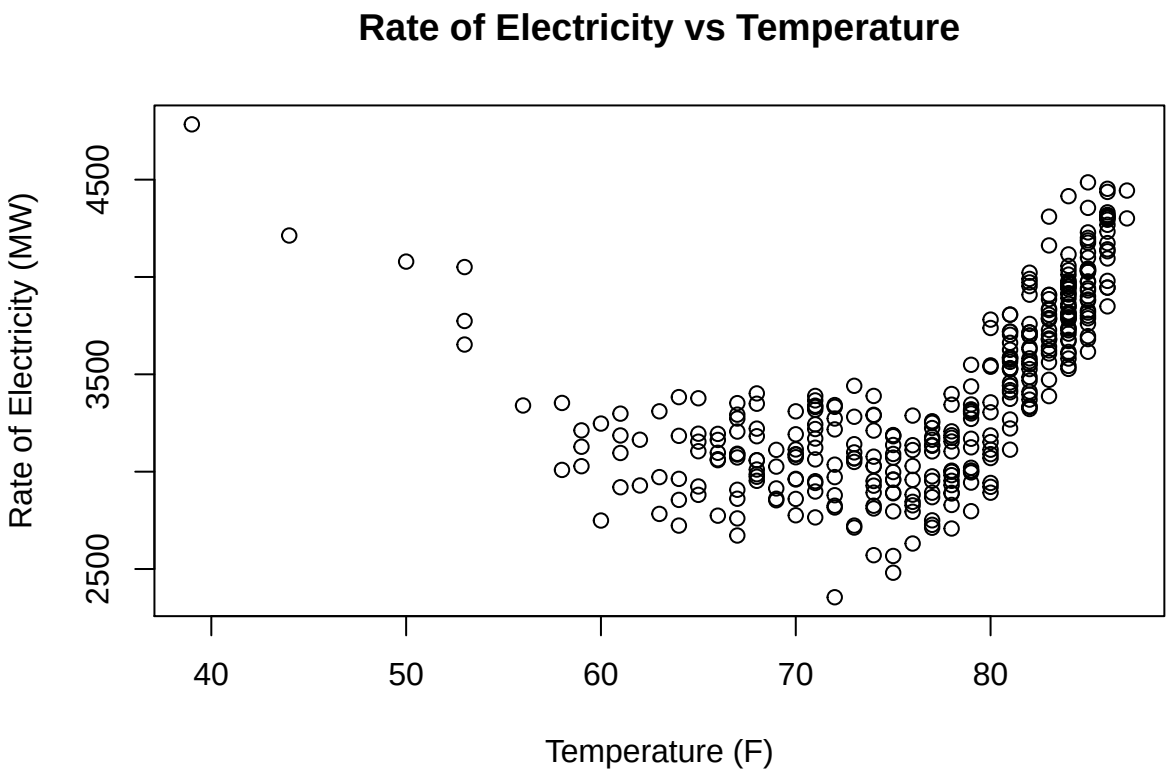
```
# time series
plot(elec$MW, type='l', xaxt='n', xlab='Day', ylab = 'Rate of Electricity (MW)',
     main='Rate of Electricity Time Series')
axis(1, at=(0:7)*52, labels=((0:7)*52))
```




```
# autocorrelation
acf(elec$MW, lag.max=365)
```




```
# MW vs temp
plot(elec$temp, elec$MW, xlab = 'Temperature (F)', ylab='Rate of Electricity (MW)',
     main='Rate of Electricity vs Temperature')
```

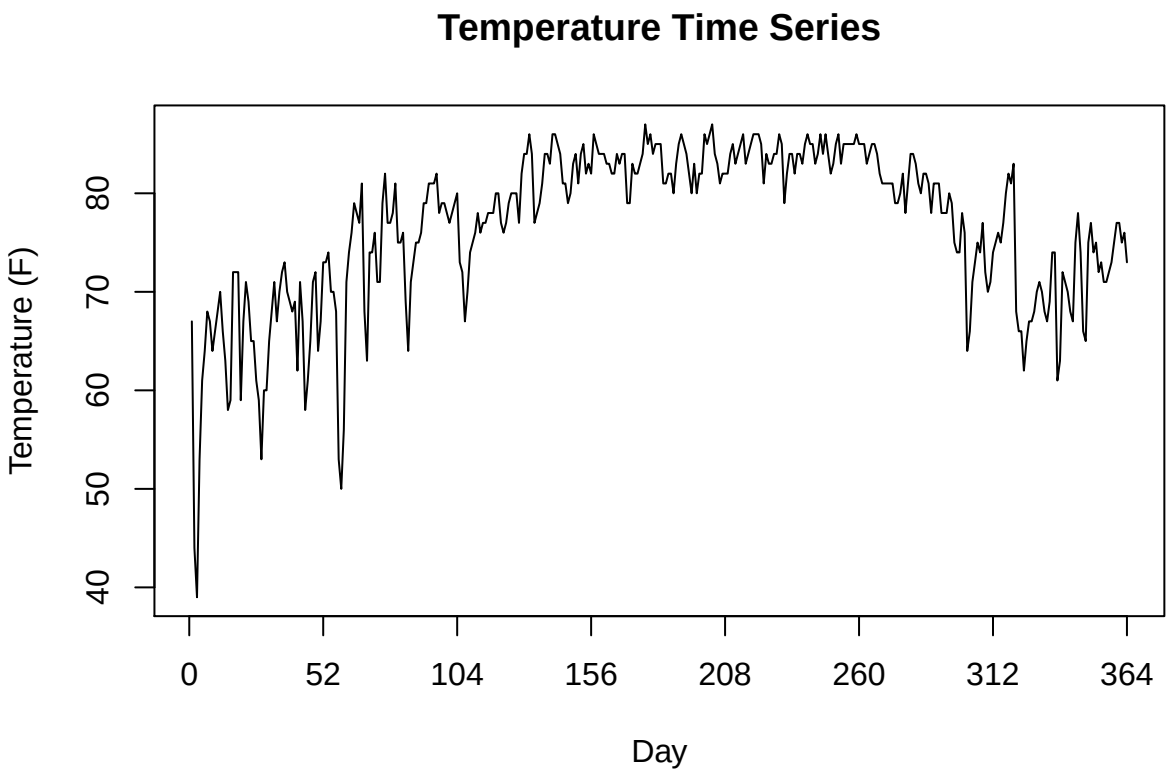


There are some interesting results from the visualizations. The time series appears to have very noticeable peaks, as well as some degree of periodicity. The ACF plot reveals a period frequency of 280-300 days for each cycle. I think this will be useful if I elect to construct an autoregressive model for MW.

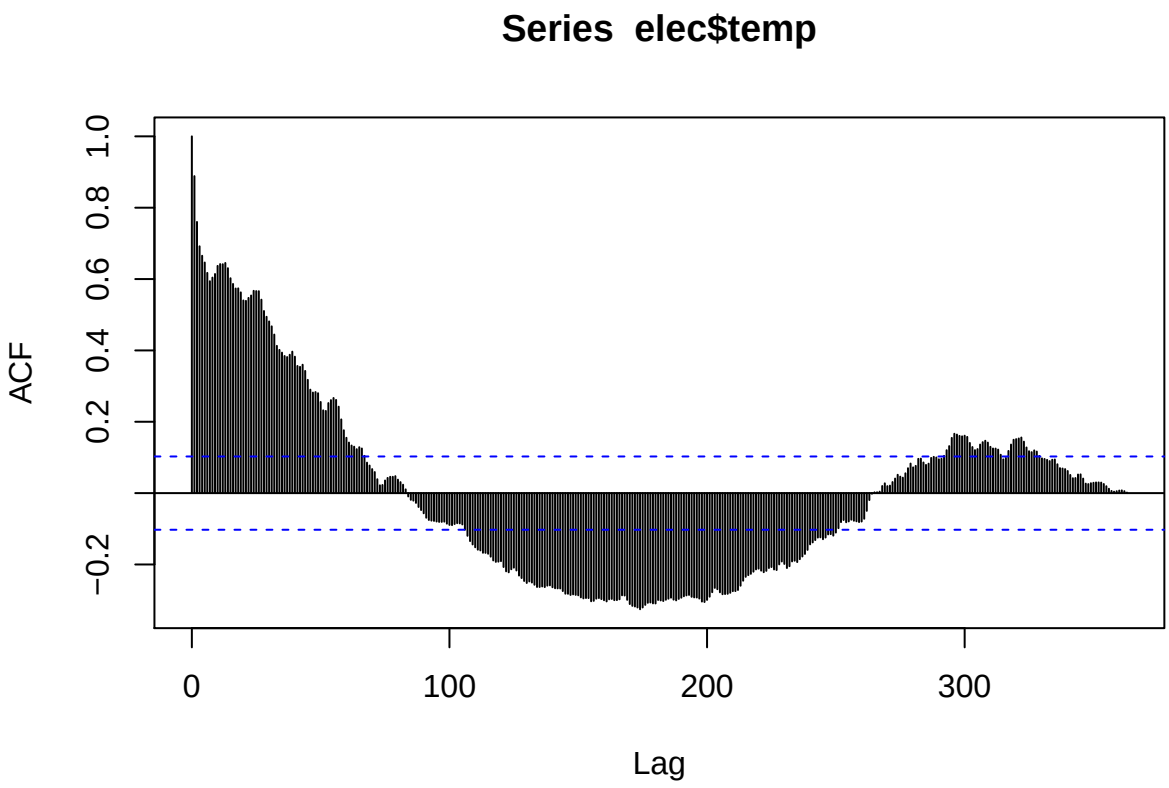
Additionally, there does appear to be a relationship for the rate of electricity with temperature. The rate of electricity has a relativley strong negative correlation (albeit for a few high leverage data points) in the temperature range of about 40-53. Then there seems to be no correlation between the variables in the range of 60-74. Lastly, there is a strong positive linear relationship for temperatures in the range of about 75-87 (87 being the maximum temperature recording in the dataset). There is clearly an interaction with certain temperature ranges, and the associated MW values.

Lastly, I want to see a time series for the *temp* variable.

```
plot(elec$temp, type='l', xaxt='n', xlab='Day', ylab = 'Temperature (F)',
     main='Temperature Time Series')
axis(1, at=(0:7)*52, labels=((0:7)*52))
```




```
acf(elec$temp, lag.max=365)
```



The temperature seems to follow a similar cycle of about 300 days.

Model Building and Forecasting Strategy I believe I have two choices for an optimal model for forecasting.

1: I construct an autoregressive model for the *MW* variable. Since they data is in chronological order it is a daily time series representing 364 days. Using the *temp* variable, sinusoidal frequency terms, and perhaps dummy variables for days of the week, I can fit an autoregressive model. Forecasting would itself require another autoregressive model for temperature, the temperature variables for day=365, and the whole week of the following year.

2. I construct a multiple regression model simply using *temp* and it's interaction with dummy variables that will represent the temperature ranges. Clearly there will be 3 tightly fit regression lines as shown in the *MW* vs *temp* plot. However, I will also need an autoregressive for the *temp* variable in order to forecast temperatures for the multiple regression model.

I think I will try both of these models. I will first do an autoregressive model for the rate of electricity *MW*. This will take longer to get results as there are more features to explore and add. Then, I will create my second option, a multiple regression model. Once both of these models are created, I will compare them and decide which to use for forecasting. Once my decision is made, I will need one more model that is an autoregressive model to forecast future *temp* values.

Let's begin with the *MW* autoregressive model.

Analysis: Autoregressive Model for MW values

I will first view the results of building an autoregressive model for the rate of electicity MW. This model will have the form:

$$MW_t = \beta_0 + \beta_1 MW_{t-1} + \epsilon_t, \epsilon_t \sim N(0, \sigma^2)$$

I am modeling the MW value at time t using the previous MW observation. The fitted value at time t will have errors. The errors should be normally distributed on mean 0 with a constant variance if they meet the residual assumptions of normality with mean 0 and a constant variance. My goal is for $|\beta_1| < 1$, to give a mean reverting model. If this condition is not met, then my series will either be a random walk ($|\beta_1| = 1$) or the values will explode ($|\beta_1| > 1$). An exploding model will not be useful for forecasting.

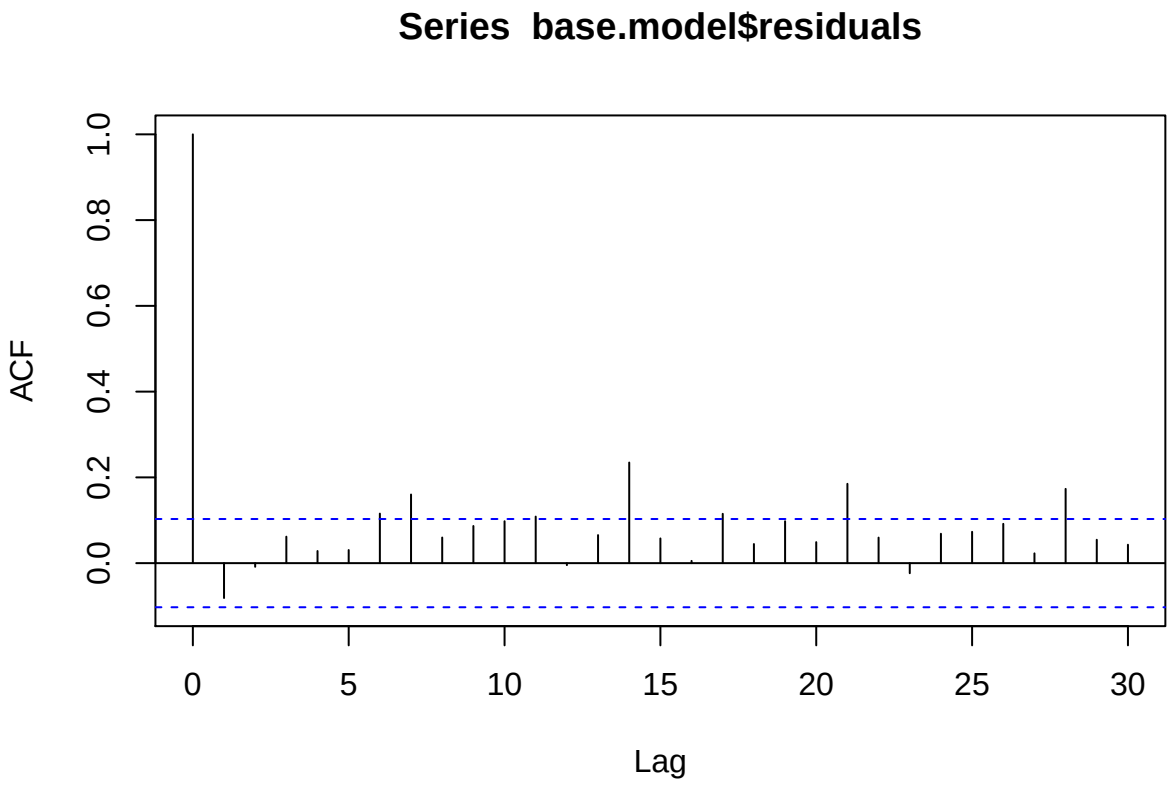
My hypotheses for this base model at the 95% level are: $H0 : |\beta_1| \geq 1$ $H1 : |\beta_1| < 1$

```
time.l1 <- 2:nrow(elec)
base.model <- lm(MW[time.l1] ~ MW[time.l1-1], data=elec)
summary(base.model)
```

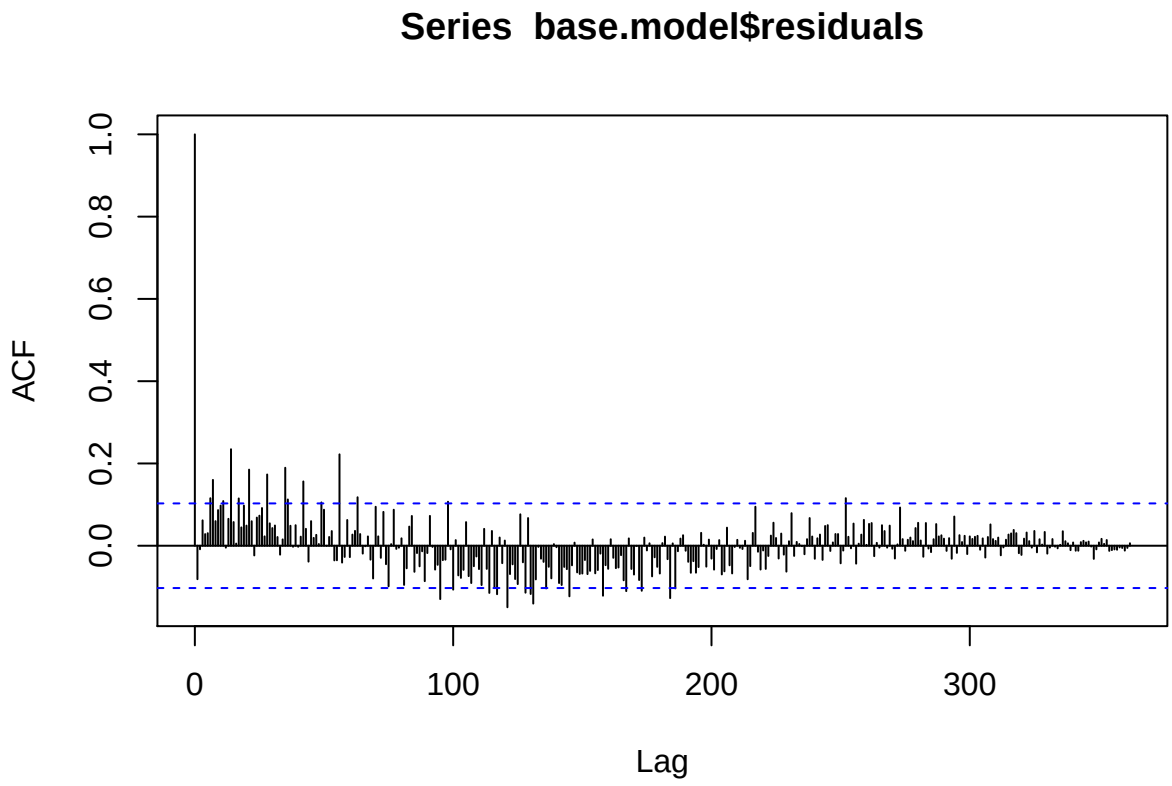
```
##
## Call:
## lm(formula = MW[time.l1] ~ MW[time.l1 - 1], data = elec)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -925.5 -198.5   -7.8  179.5 1326.7
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     986.62784  126.48658    7.80 6.71e-14 ***
## MW[time.l1 - 1]   0.71098    0.03678   19.33  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 318.8 on 361 degrees of freedom
## Multiple R-squared:  0.5087, Adjusted R-squared:  0.5073
## F-statistic: 373.8 on 1 and 361 DF,  p-value: < 2.2e-16
```

The summary output tells me that the null hypothesis should be rejected with $\beta_1 = 0.71098$. With an extremley small p-value, the autoregressive model is mean reverting. Additionally, just using the previous day's rate of electricity is able to explain about 50% of the variance in the next day rate of electricity. To confirm these findings, I must check the resiudal assumptions.

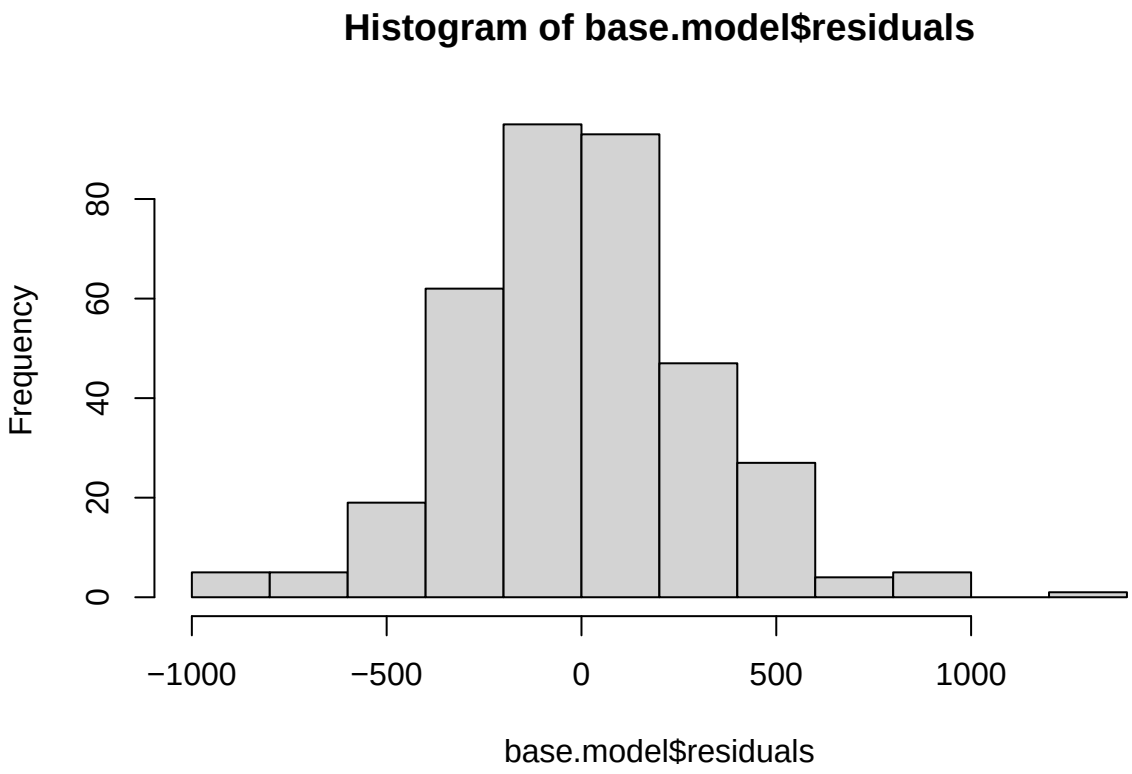
```
acf(base.model$residuals, lag.max = 30)
```




```
acf(base.model$residuals, lag.max = 364)
```



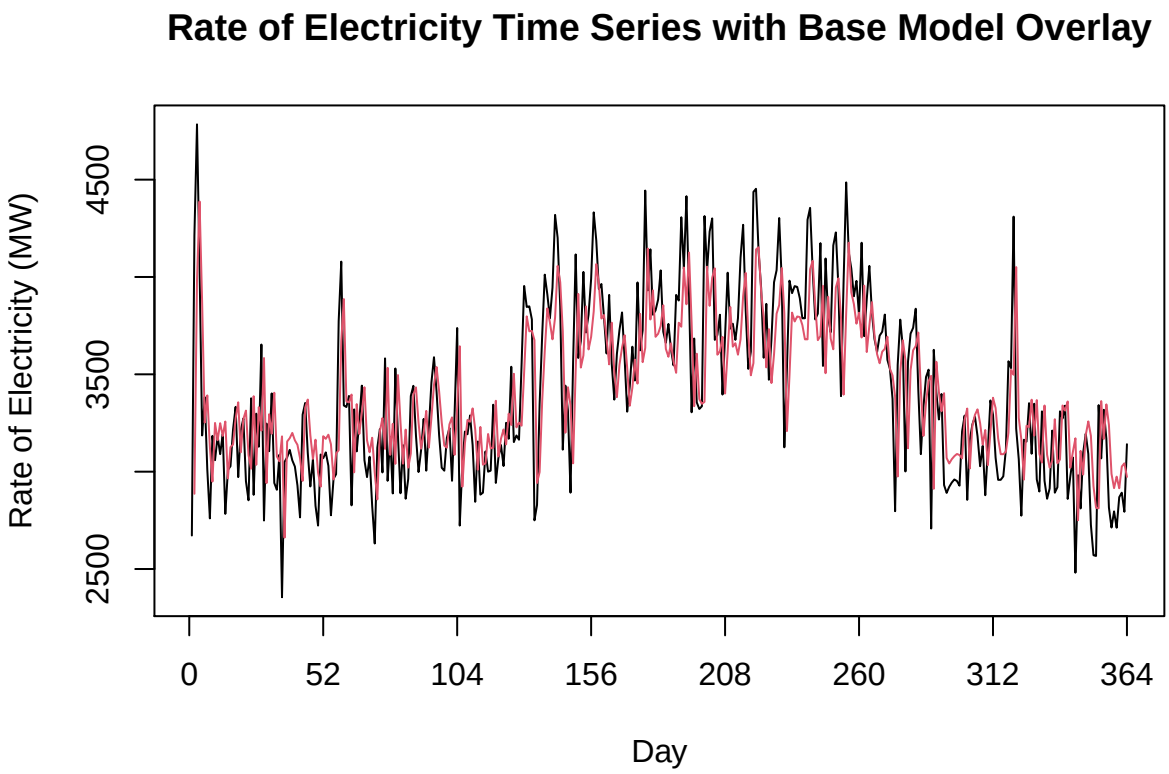

```
hist(base.model$residuals)
```



The first autocorrelation plot of the resiudals (which is the autoregressive method for checking residual assumptions) show that there are not many points with a signficant amount of leftover correlation. The second also appears to be okay, but can be improved. The histogram also shows a normal distribution centered on 0, with the only noteable issue being an outlier (or possibly a few) on the right side of the histogram. I think that the residual assumptions are met, and I will confirm my rejection of the null hypthesis and accept the alternative H_1. This is a mean reverting autoregressive model.

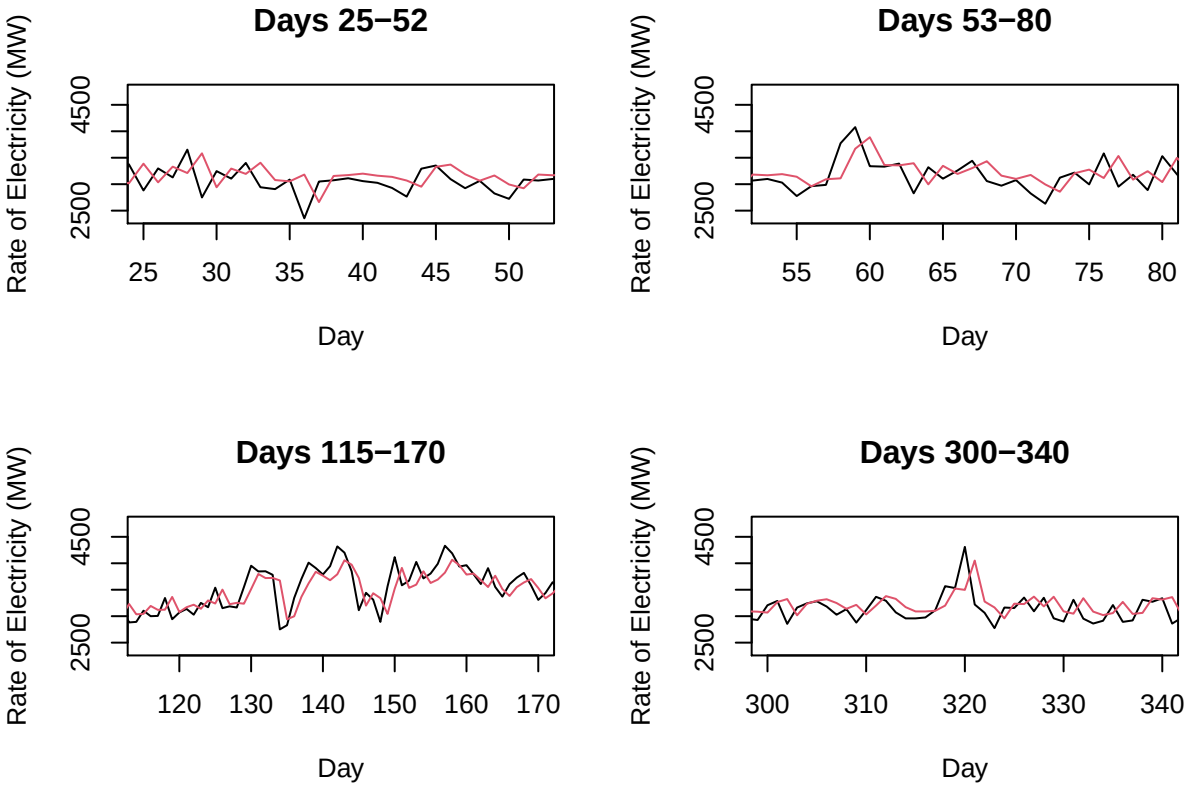
Let's see the original and fitted time series.

```
plot(elec$MW, type='l', xaxt='n', xlab='Day', ylab = 'Rate of Electricity (MW)',
     main='Rate of Electricity Time Series with Base Model Overlay', xlim=c(1,nrow(elec)))
axis(1, at=(0:7)*52, labels=((0:7)*52))
#adding line of fitted values
lines((time.l1), base.model$fitted.values, col=2)
```



It can be seen that overall, the model does a relatively good job of fitting the time series. However, it is far from perfect, and certain peaks and valleys in the time series are not fit as well as I would like.

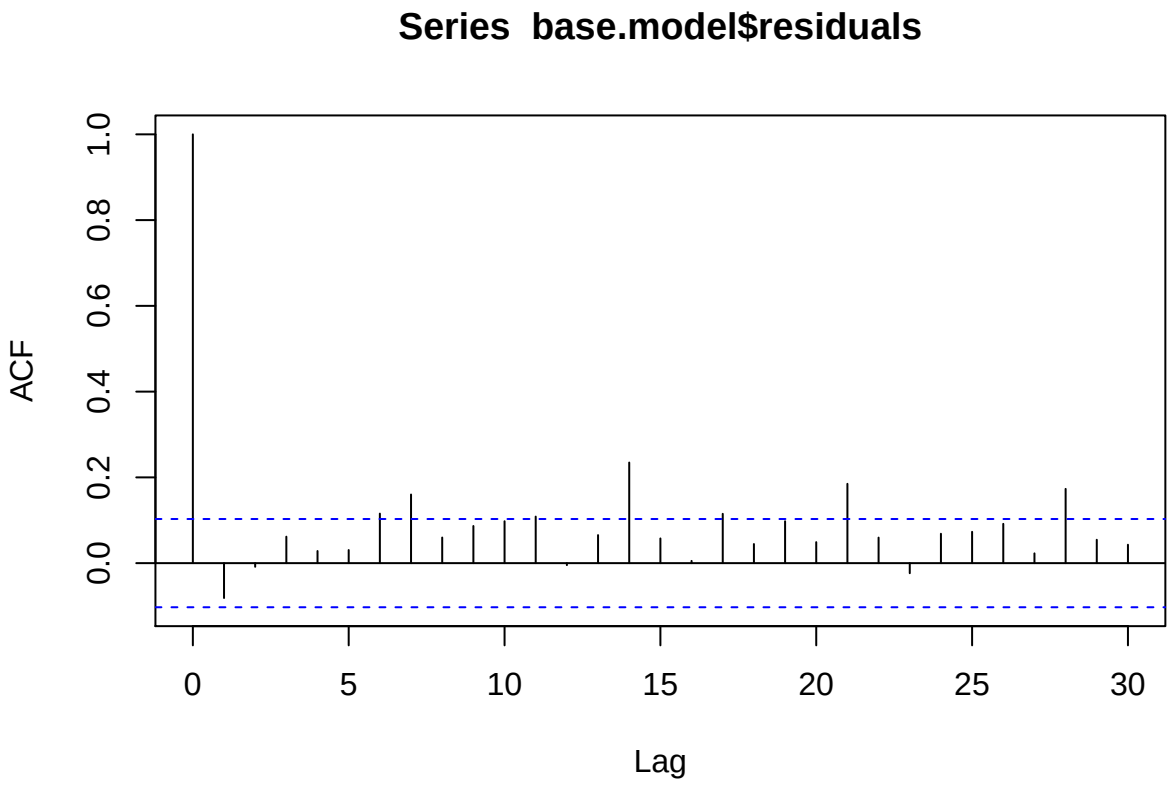
```
par(mfrow=c(2,2))
plot(elec$MW, type='l', xlab='Day', ylab = 'Rate of Electricity (MW)',
     main='Days 25-52', xlim=c(25,52))
lines((time.l1), base.model$fitted.values, col=2)

plot(elec$MW, type='l', xlab='Day', ylab = 'Rate of Electricity (MW)',
     main='Days 53-80', xlim=c(53,80))
lines((time.l1), base.model$fitted.values, col=2)

plot(elec$MW, type='l', xlab='Day', ylab = 'Rate of Electricity (MW)',
     main='Days 115-170', xlim=c(115,170))
lines((time.l1), base.model$fitted.values, col=2)

plot(elec$MW, type='l', xlab='Day', ylab = 'Rate of Electricity (MW)',
     main='Days 300-340', xlim=c(300,340))
lines((time.l1), base.model$fitted.values, col=2)
```



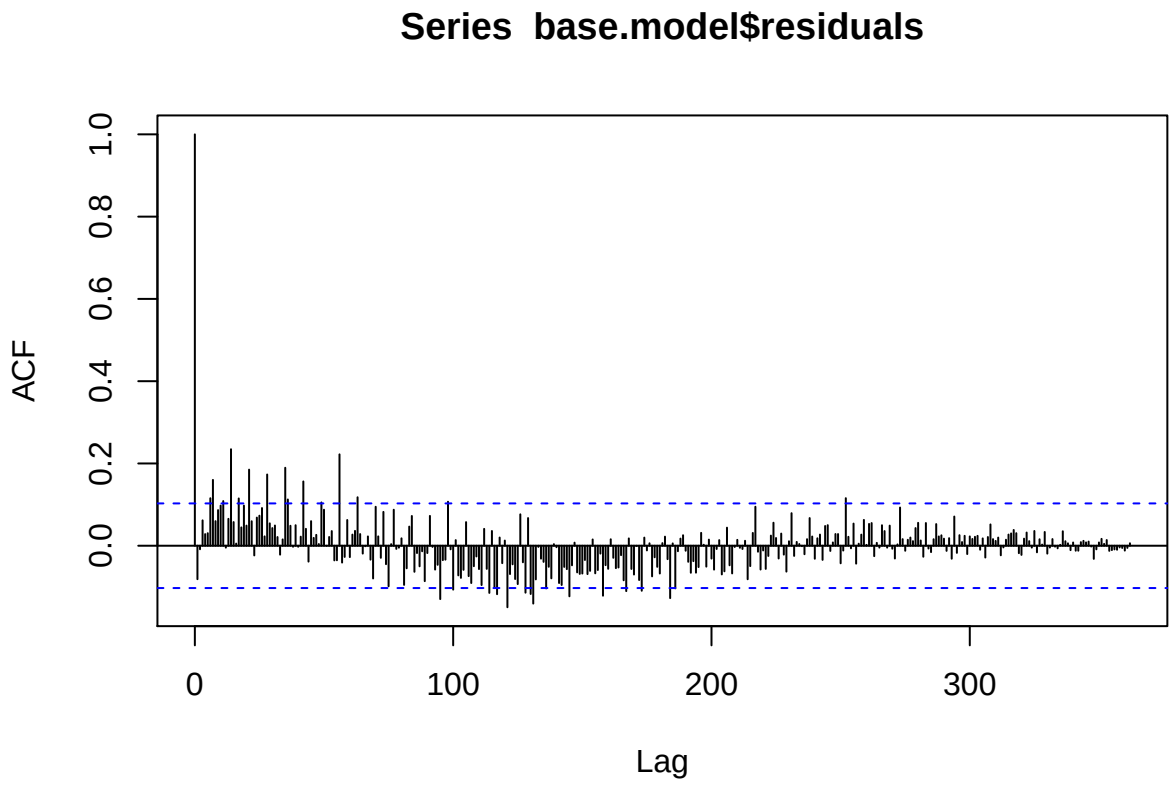
The plots above show an unsatisfactory amount of underprediction for the valleys and the peaks of the time series. Peak electric rate is critical for correct prediction in the real context of modeling electricity usage. It is also a topic of interest in this research. Additionally, the fitted MW values are not able to accurately capture the sudden changes in MW and are 'slow' to respond to these changes of direction at each time step with a 1 step delay. This results in an alm

It was previously mentioned that the ACF plots of the residuals (showing 'leftover' correlation) had some very interesting results. Let's take a look at these plots again.

```
acf(base.model$residuals, lag.max = 30)
```




```
acf(base.model$residuals, lag.max = 364)
```

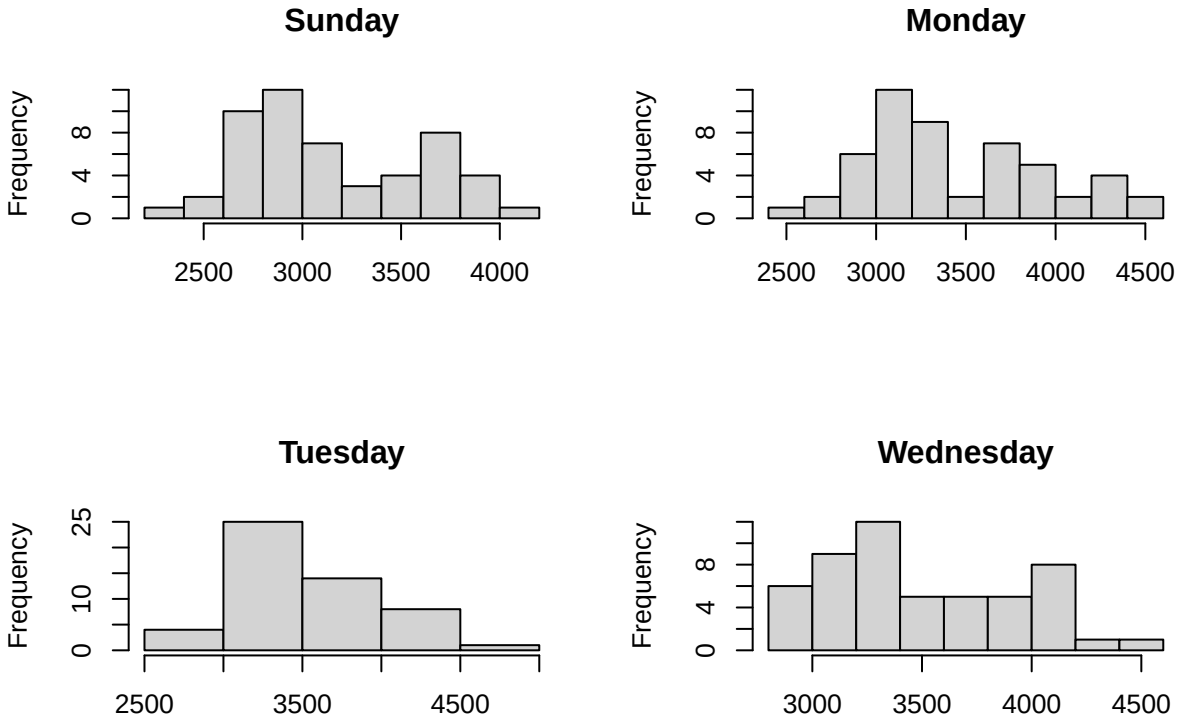


What is curious to me is an apperent 7 day cycle in the plot showing 30 days of autocorrelation of the model's residuals. Every 7th residual has autocorrelation above the signficance line, indicating poorly fit observations. This will be explored in the next section when features are added to build a more accurate model.

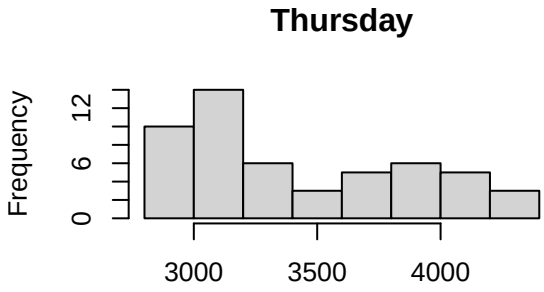
Analysis: Exploring and adding extra features

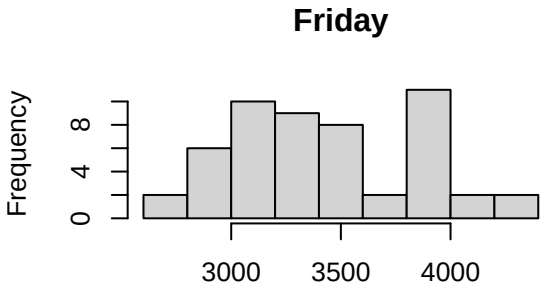
I will start by viewing a histogram of rates of electricity *MW* for each of the days of the week. When I loaded my data I added a variable for which day of the week it is (Sunday through Saturday) as my intuition told me that this is likely to impact the rate of electricity. Let's take a look.

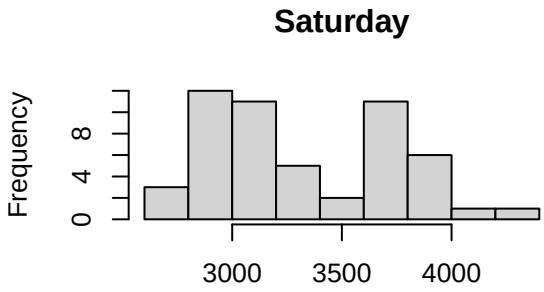
```
par(mfrow=c(2,2))
hist(elec$MW[elec$DOW=='Sun'], main = 'Sunday', xlab = '')
hist(elec$MW[elec$DOW=='Mon'], main = 'Monday', xlab = '')
hist(elec$MW[elec$DOW=='Tue'], main = 'Tuesday', xlab = '')
hist(elec$MW[elec$DOW=='Wed'], main = 'Wednesday', xlab = '')
```




```
par(mfrow = c(2,2))
hist(elec$MW[elec$DOW=='Thur'], main = 'Thursday', xlab = '')
hist(elec$MW[elec$DOW=='Fri'], main = 'Friday', xlab = '')
hist(elec$MW[elec$DOW=='Sat'], main = 'Saturday', xlab = '')
```







I also want to take a look at what days the peak electric values occur on.

```
# function taken from https://stackoverflow.com/a/18451329
whichpart <- function(x, n=30) {
  nx <- length(x)
  p <- nx-n
  xp <- sort(x, partial=p)[p]
  which(x > xp)
}
# indices of top 50 values
indices <- whichpart(x = elec$MW, n=49)
# converting vector to table and then to frame to view frequencies for each day
as.data.frame(table(elec[indices, 3]))
```

```
##   Var1 Freq
## 1  Fri    6
## 2  Mon   11
## 3  Sat    3
## 4  Sun    1
## 5 Thur    8
## 6  Tue   10
## 7  Wed   10
```

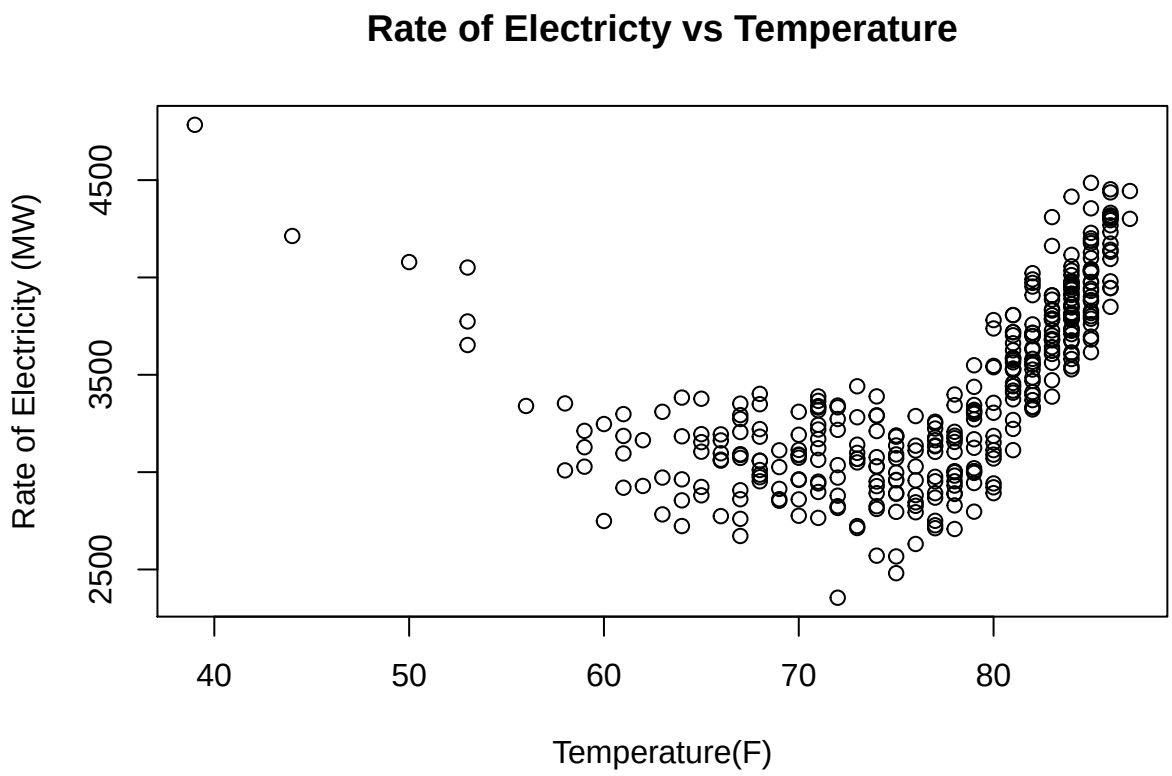
The histograms reveal that weekdays generally have a higher frequency of large MW values, and the maximums appear to be smaller as well. The frequency table above has an even more impactful finding in this regard. Of the 49 maximum values of MW, Saturday and Sunday have a combined frequency of only 4.

Compared to an expecation of 14 under the assumption that temperature (which was earlier shown to be correlated with MW) is not itself affected by the day of the week (which is a reasonable assumption since days of the week are arbitrary names for a societal structure), I stand to believe that this is not a random occurence. Additionally; Friday, Saturday, and Sunday have a combined frequency of 10 vs an expecation of 21.

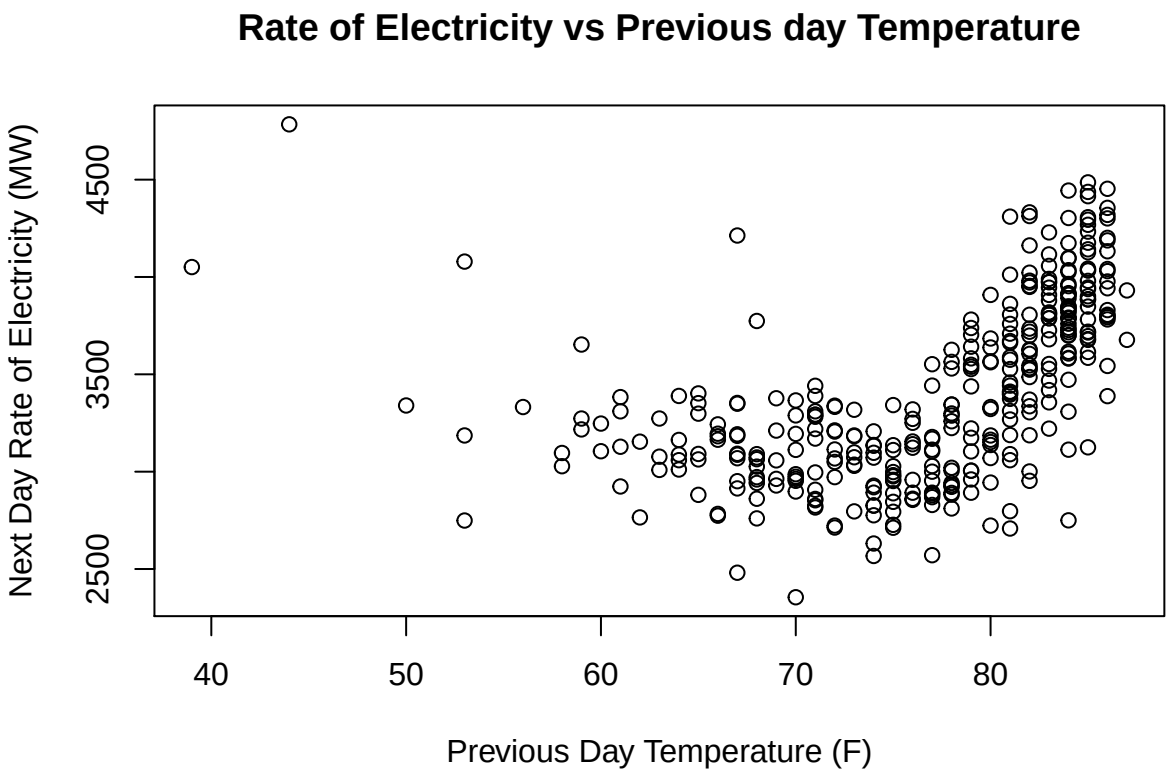
These findings are promising for further models, since 2 dummy variables can be created for weekends and weekdays or Friday-Sunday and Monday-Thursday rather than indivudal dummies for each day of the week; this allows a better predictive variance and a less biased model reducing the possibility of over fitting.

The other important feature for this model is temperature. Let's view the plot of rates of electricity vs the temperatures again, as well as rates of electricity vs temperature from the previous day, a histogram, and a boxplot.

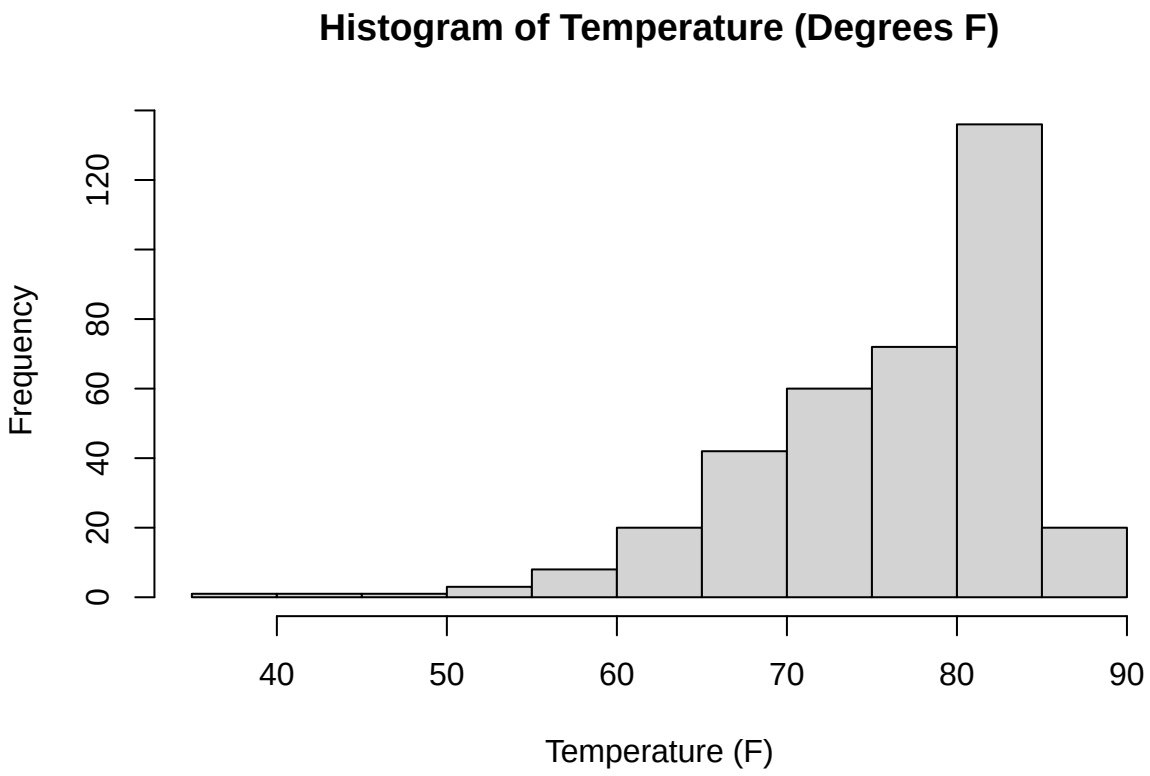
```
plot(elec$temp, elec$MW, main = 'Rate of Electricty vs Temperature',
     xlab='Temperature(F)', ylab='Rate of Electricity (MW)')
```



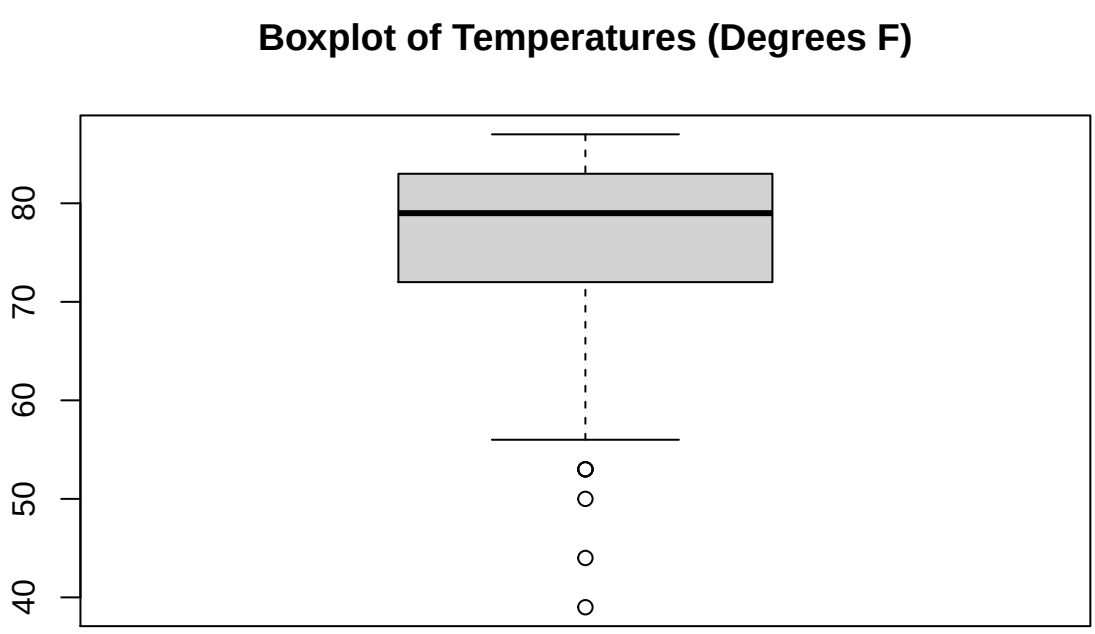

```
plot(elec$temp[time.l1-1], elec$MW[time.l1],
     main = 'Rate of Electricity vs Previous day Temperature',
     xlab='Previous Day Temperature (F)', ylab='Next Day Rate of Electricity (MW)')
```




```
hist(elec$temp, xlab = 'Temperature (F)', main = 'Histogram of Temperature (Degrees F)')
```




```
boxplot(elec$temp, main='Boxplot of Temperatures (Degrees F)')
```



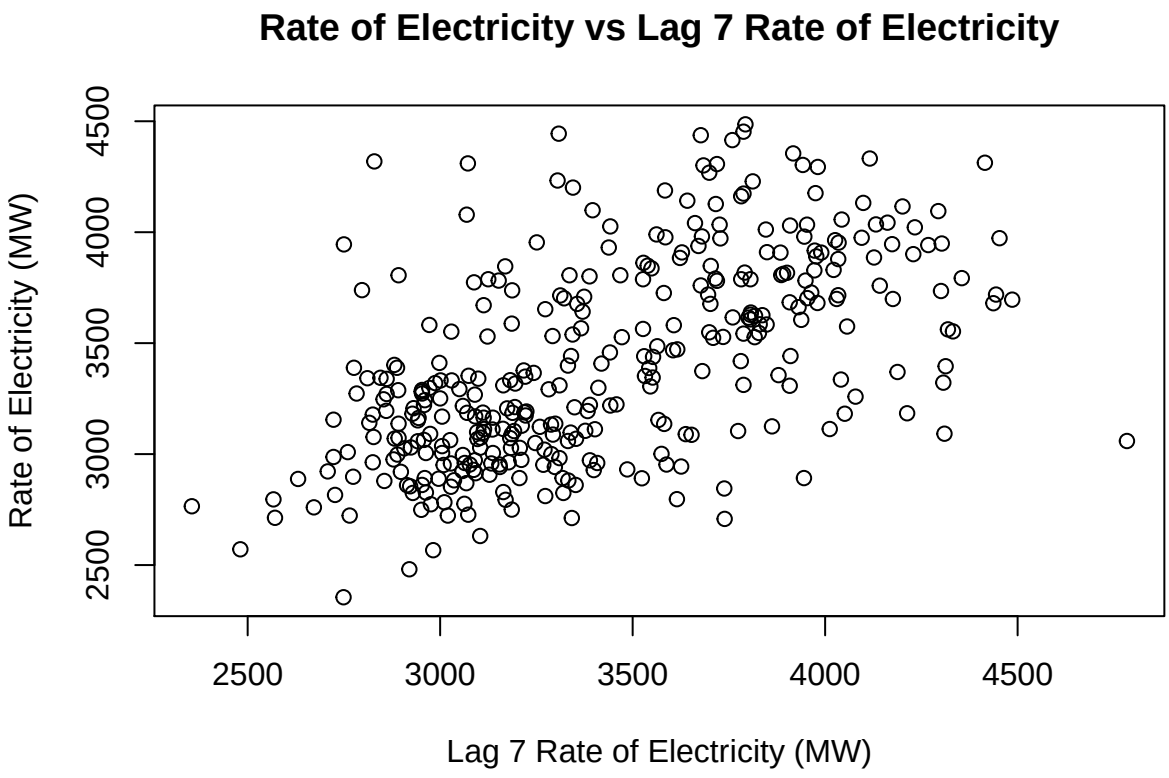
The first plot shows that there is a pattern associated with a day's MW value given a temperature. The peak electricity usage happens on the more extreme temperature days as a general rule. It can also be seen that the moderate temperatures have a somewhat consistent range of electricity usage that will be associated with them.

Peak electric use soley happens on the temperature extremities. In the temperature range of 55-75 there is not a single outlier for peak electricity usage. I believe that if I leverage this fact and create a dummy variable for indication of temperature extremes, then I can much more accurately capture the spikes or dips in the MW values.

The histogram reveals a distribution that is heavily skewed left for the temperature values. About 95% of the data is in the range of 60-85 degrees. The temperatures outside of this range are extremes for the region. The boxplot solidifies these findings.

The final point of interest for me is to see if a lag=7 term has a better correlation than the lag=1 term I have been working with.

```
time.l7 <- 8:nrow(elec)
plot(elec$MW[time.l7-7], elec$MW[time.l7], xlab='Lag 7 Rate of Electricity (MW)',
     ylab='Rate of Electricity (MW)', main='Rate of Electricity vs Lag 7 Rate of Electricity')
```



I will not investigate this further as the plot shows a much weaker correlation for the lag=7 term than my original lag=1 term.

I will now have a main frame for training my autoregressive model called *model1*. After adding all of my features, I can select a model that I will then use to forecast the MW values for the last day of the year (day 365), and the first week of the following year if I elect to use this autoregressive model over my not yet built multiple regression model. However, this frame can be used for my later temperature autoregressive model that I require for forecasting MW values.

```
# creating features
# showing the time perioid one more time
time.l1 <- 2:nrow(elec)

# sinusoidal function for the yearly cycle of temperatures using
#sin(2pi*t/k) and cos(2pi*t/k), with k=300 for my 300 day periodicity in temperature
sin300 <- sin(2*pi*time.l1/300)
cos300 <- cos(2*pi*time.l1/300)

# lag 1 rates of electricity (MW)
AR1 <- elec$MW[time.l1-1]

# MW values
MW <- elec$MW[time.l1]

#current day temperatures
cur.temp <- elec$temp[time.l1]
```

```r
#previous day temperature
prev.temp <- elec$temp[time.l1-1]

# dummy variable for more extreme temp values
temp.extreme <- rep(FALSE, nrow(elec))
temp.extreme[elec$temp >= 85] <- TRUE
temp.extreme[elec$temp <= 55] <- TRUE
temp.extreme <- as.factor(temp.extreme)
temp.extreme <- temp.extreme[time.l1]
# creating dummy variables for temeprature ranges of the current day
temp.range <- rep('60<temp<80', nrow(elec))
temp.range[elec$temp <= 60] <- 'temp<=60'
temp.range[elec$temp >= 80] <- 'temp>=80'
temp.range <- as.factor(temp.range)
temp.range <- temp.range[time.l1]

# creating dummy variables for weekends (which will be Saturday, Sunday)
weekend <- rep(FALSE, nrow(elec))
weekend[elec$DOW == 'Sun'] <- TRUE
weekend[elec$DOW == 'Sat'] <- TRUE
weekend <- as.factor(weekend)
weekend <- weekend[time.l1]

# new frame
train <- data.frame(
  MW=MW,
  AR1 = AR1,
  temp.range = temp.range,
  cur.temp = cur.temp,
  weekend = weekend,
  sin300 = sin300,
  cos300 = cos300,
  temp.extreme = temp.extreme
  )
```

For my first model, I want to check the performance of an autoregressive term, the sinusoidal variables, and the dummy variable for weekends.

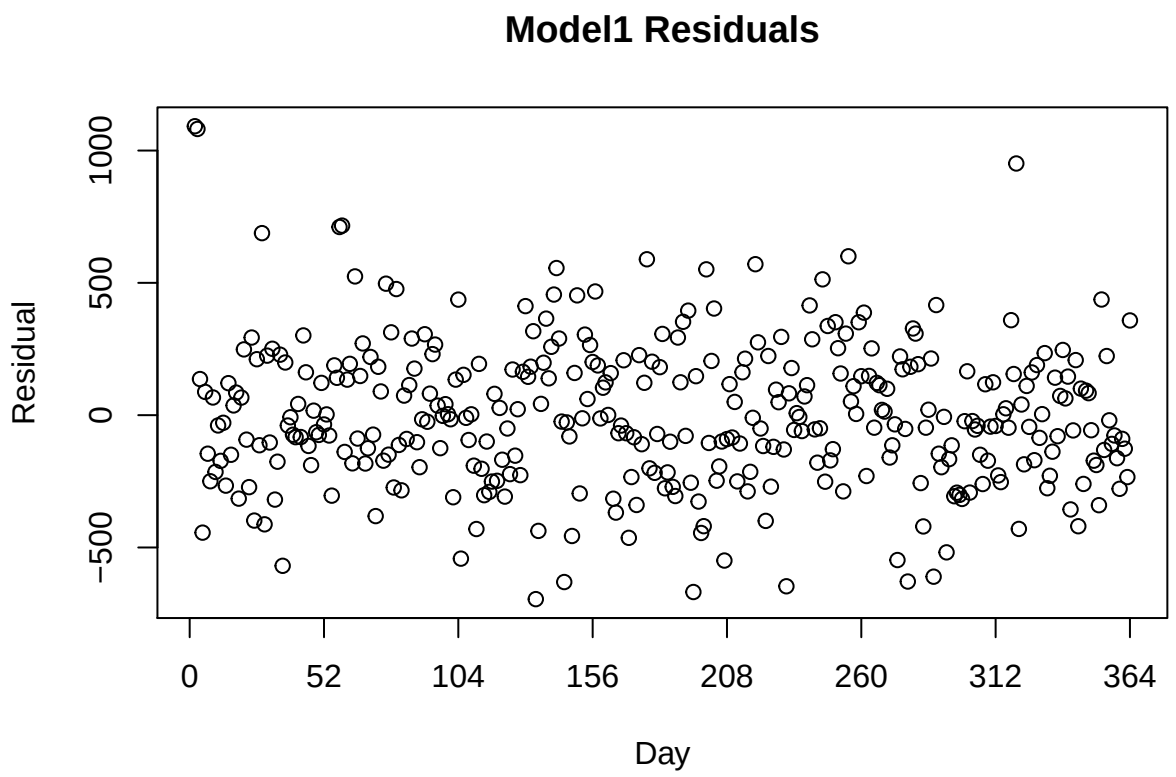
```r
model1 <- lm(MW~AR1 + cos300 + sin300 + weekend, data=train)
summary(model1)
```

```
##
## Call:
## lm(formula = MW ~ AR1 + cos300 + sin300 + weekend, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -695.21 -174.31  -22.32  169.16 1091.87
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2210.68964  159.84588  13.830  < 2e-16 ***
## AR1            0.38086    0.04567   8.340 1.63e-15 ***
```

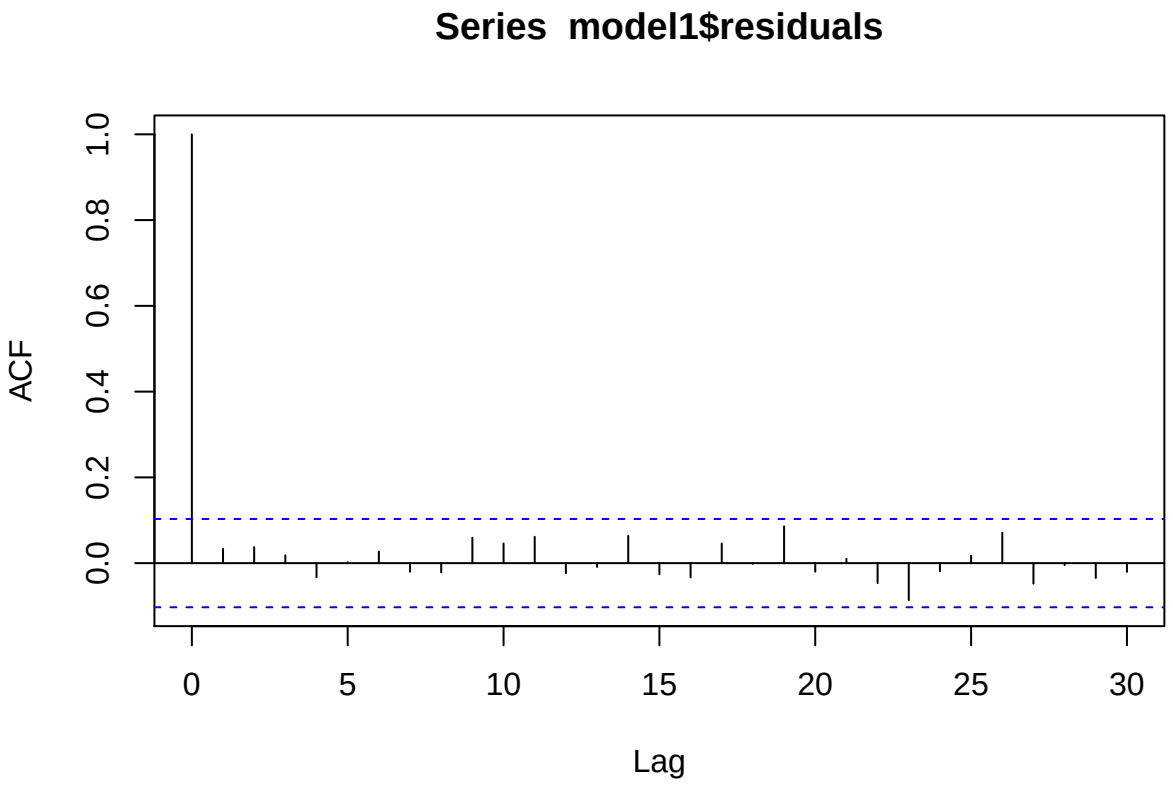
```
## cos300       -96.03488   21.44990  -4.477 1.02e-05 ***
## sin300      -268.71266   28.90769  -9.296  < 2e-16 ***
## weekendTRUE -208.20493   31.76091  -6.555 1.94e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 272.6 on 358 degrees of freedom
## Multiple R-squared:  0.6439, Adjusted R-squared:  0.6399
## F-statistic: 161.8 on 4 and 358 DF,  p-value: < 2.2e-16
```

Resiudals:

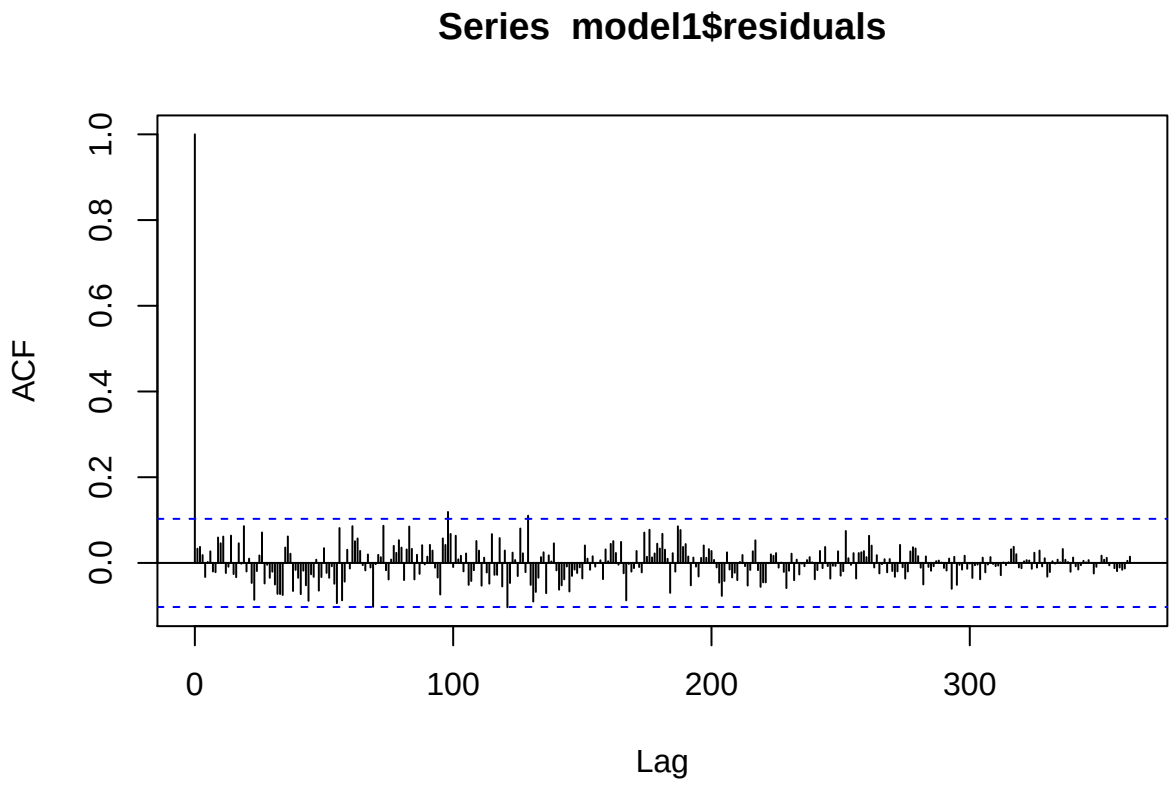
```
plot(time.l1, model1$residuals, xaxt='n', xlab='Day',
     ylab = 'Residual', main='Model1 Residuals')
axis(1, at=(0:7)*52, labels=((0:7)*52))
```



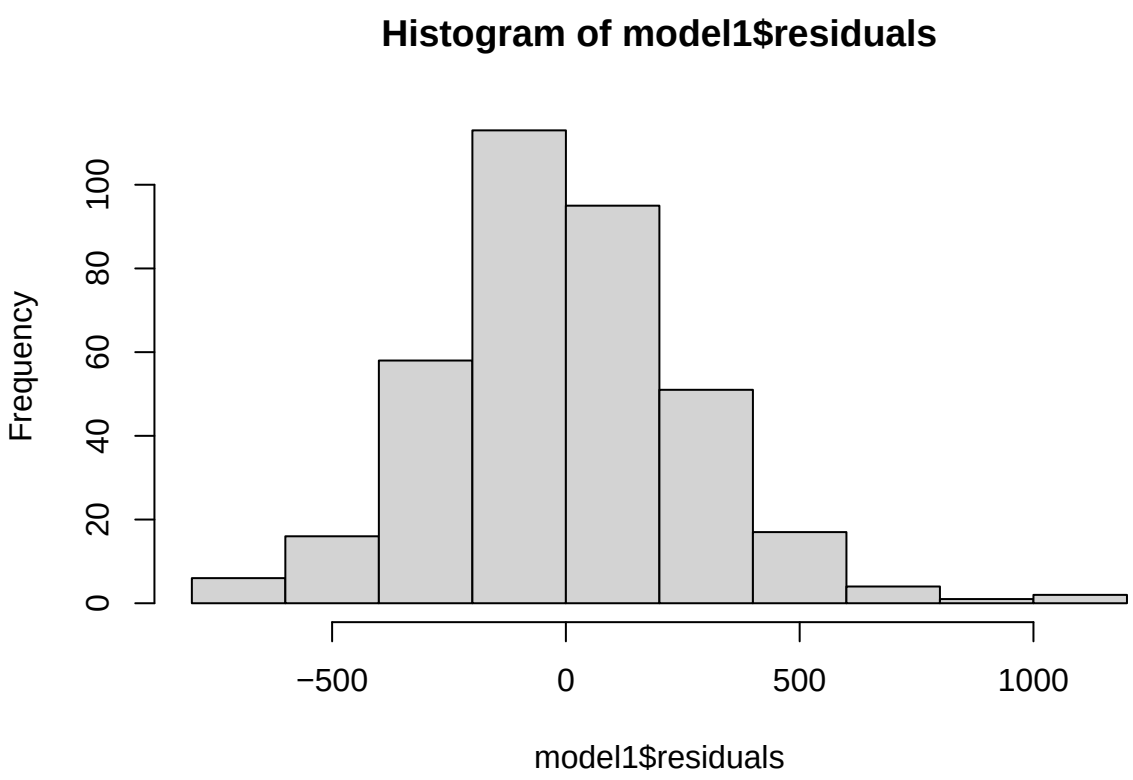

```
acf(model1$residuals, lag.max=30)
```




```
acf(model1$residuals, lag.max=365)
```




```
hist(model1$residuals)
```

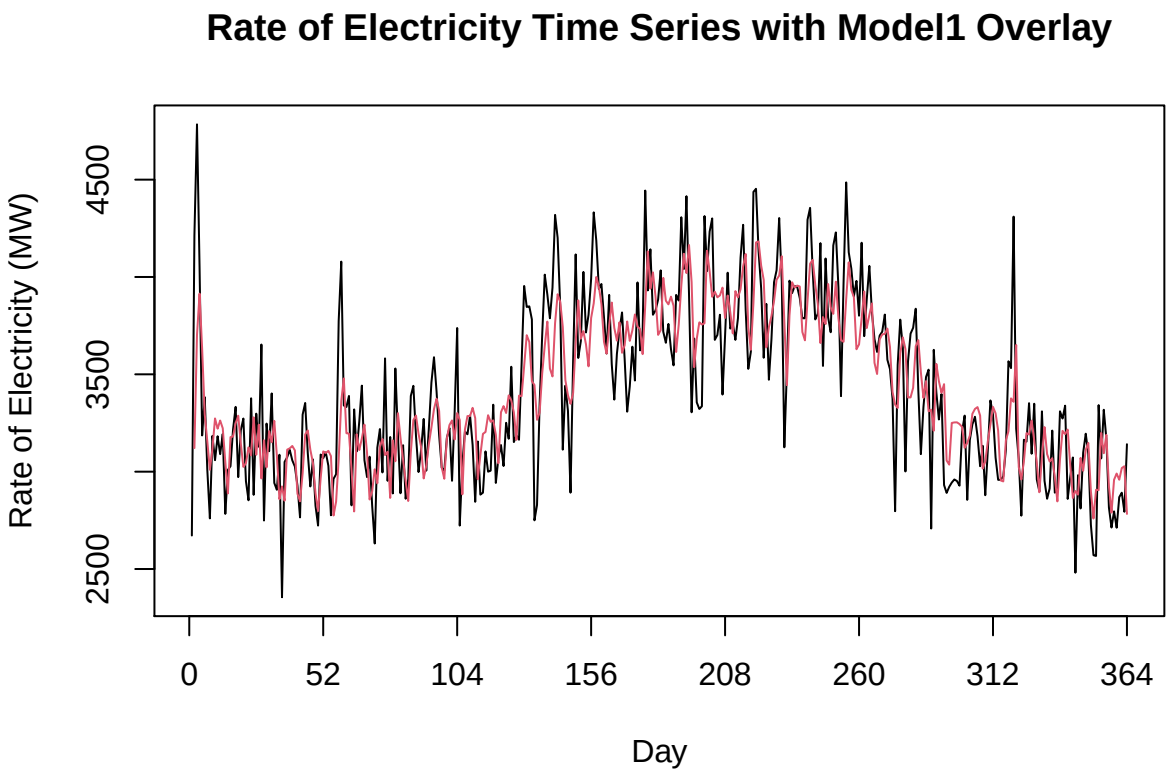


Already it appears that this model has improved on the baseline model. The explained variance in MW values has increased from about 50% to almost 65%. All regression coefficients are very significant with low p-values. The weekend dummy variable has a negative value as I expected from the exploratory analysis of the feature. The null hypothesis that the model is not mean reverting can once again be rejected, with normally distributed resiudal values having mean 0 and a constant variance.

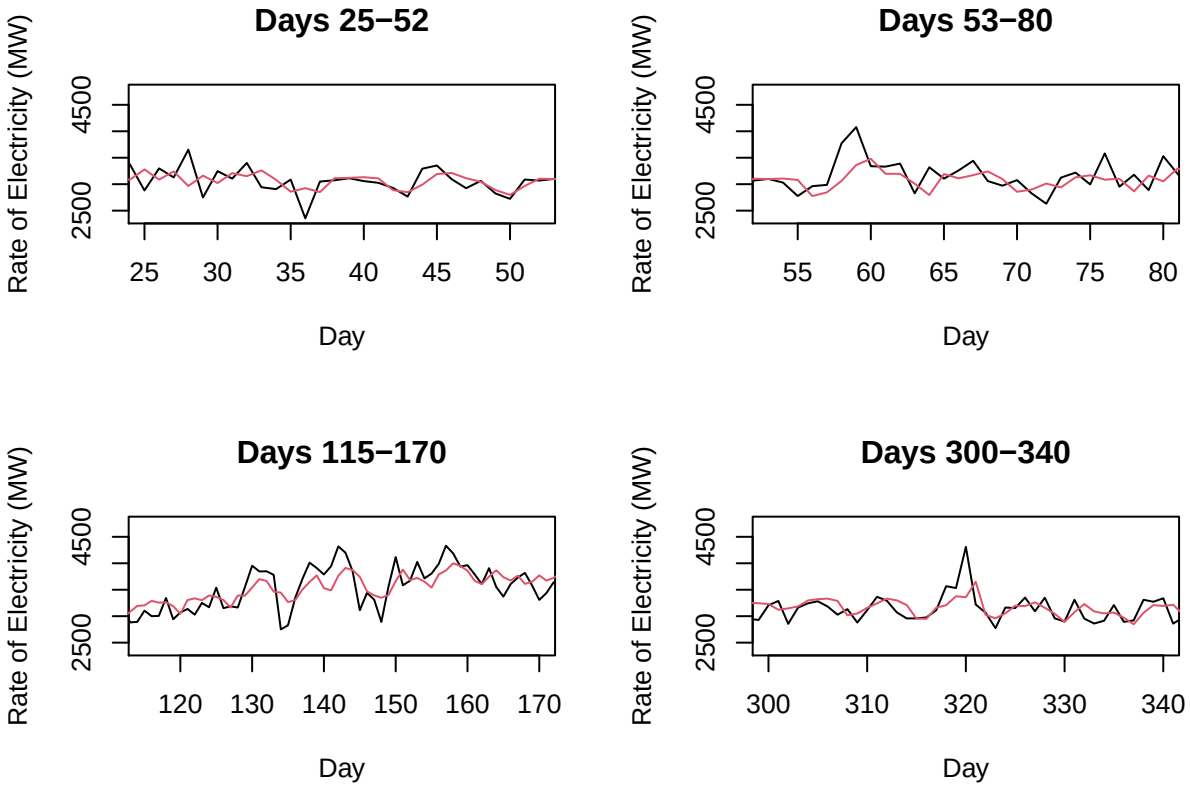
The residual plots reveal a bias for underprediciton. This must be addressed, since peak electric rates are an important component for this modeling.

The fitted time series is below.

```
plot(elec$MW, type='l', xaxt='n', xlab='Day', ylab = 'Rate of Electricity (MW)',
     main='Rate of Electricity Time Series with Model1 Overlay')
axis(1, at=(0:7)*52, labels=((0:7)*52))
#adding line of fitted values
lines((time.l1), model1$fitted.values, col=2)
```




```
par(mfrow=c(2,2))
plot(elec$MW, type='l', xlab='Day', ylab = 'Rate of Electricity (MW)',
     main='Days 25-52', xlim=c(25,52))
lines((time.l1), model1$fitted.values, col=2)

plot(elec$MW, type='l', xlab='Day', ylab = 'Rate of Electricity (MW)',
     main='Days 53-80', xlim=c(53,80))
lines((time.l1), model1$fitted.values, col=2)

plot(elec$MW, type='l', xlab='Day', ylab = 'Rate of Electricity (MW)',
     main='Days 115-170', xlim=c(115,170))
lines((time.l1), model1$fitted.values, col=2)

plot(elec$MW, type='l', xlab='Day', ylab = 'Rate of Electricity (MW)',
     main='Days 300-340', xlim=c(300,340))
lines((time.l1), model1$fitted.values, col=2)
```



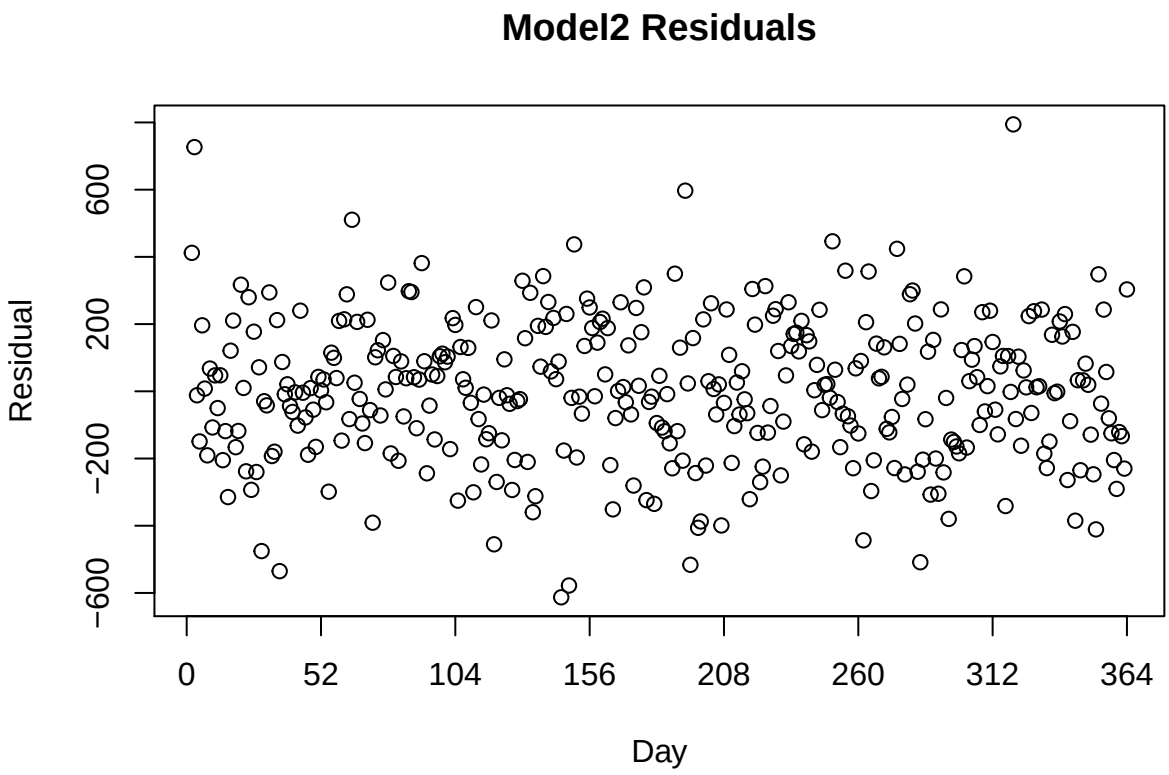
I will now fit a second model adding in temperature values which include the continous temperature values, the dummy variable for ranges of temperature values, and a dummy for indication if the temperature is an extreme range of values.

```
model2 <- lm(MW~AR1 + cos300 + sin300 + weekend + cur.temp + temp.range + temp.extreme, data=train)
summary(model2)
```

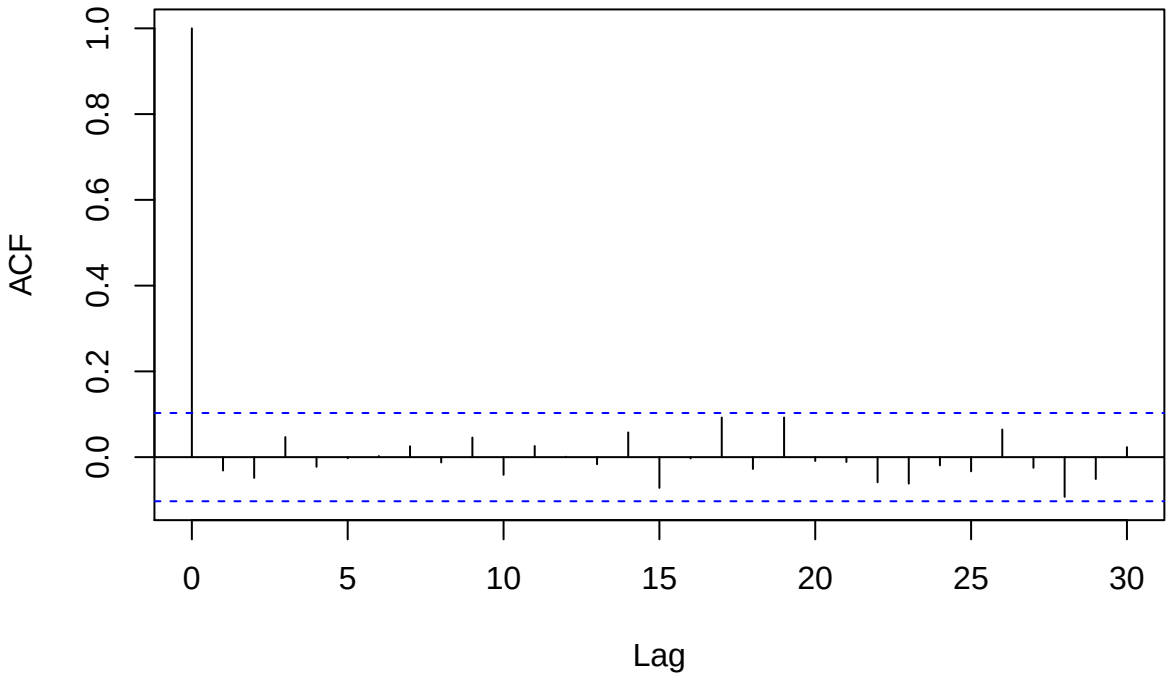
```
##
## Call:
## lm(formula = MW ~ AR1 + cos300 + sin300 + weekend + cur.temp +
##     temp.range + temp.extreme, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -612.93 -143.41    3.23  141.49  794.21
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)         3139.9907   270.7921  11.596  < 2e-16 ***
## AR1                    0.1499     0.0387   3.872 0.000128 ***
## cos300               -61.1987    20.0293  -3.055 0.002418 **
## sin300              -115.5039    29.0025  -3.983 8.28e-05 ***
## weekendTRUE         -188.8148    24.8272  -7.605 2.59e-13 ***
## cur.temp              -5.5556     3.1547  -1.761 0.079096 .
## temp.rangetemp<=60   175.3414    83.2764   2.106 0.035949 *
## temp.rangetemp>=80   410.5329    45.8639   8.951  < 2e-16 ***
```

```
## temp.extremeTRUE    395.6929    37.2090  10.634  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 212.6 on 354 degrees of freedom
## Multiple R-squared:  0.7858, Adjusted R-squared:  0.7809
## F-statistic: 162.3 on 8 and 354 DF,  p-value: < 2.2e-16
```

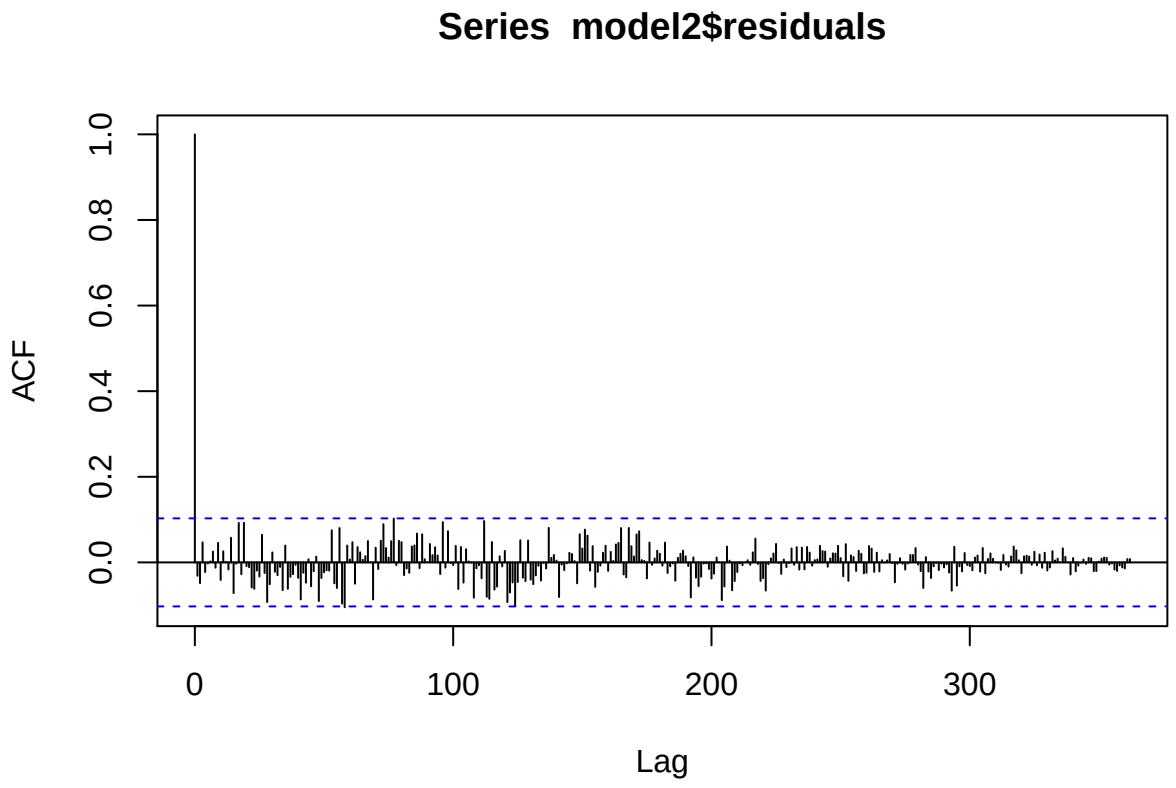
```
plot(time.l1, model2$residuals, xaxt='n', xlab='Day', ylab = 'Residual',
     main='Model2 Residuals')
axis(1, at=(0:7)*52, labels=((0:7)*52))
```



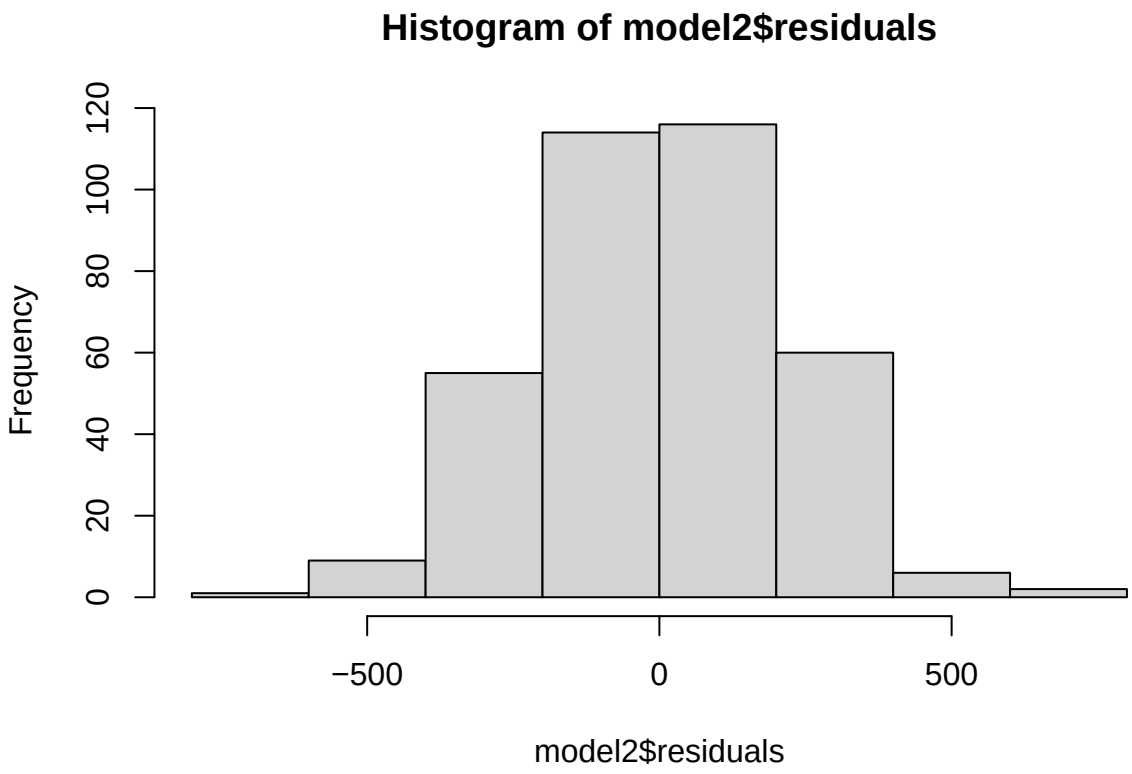

```
acf(model2$residuals, lag.max=30)
```




```
acf(model2$residuals, lag.max=365)
```




```
hist(model2$residuals)
```



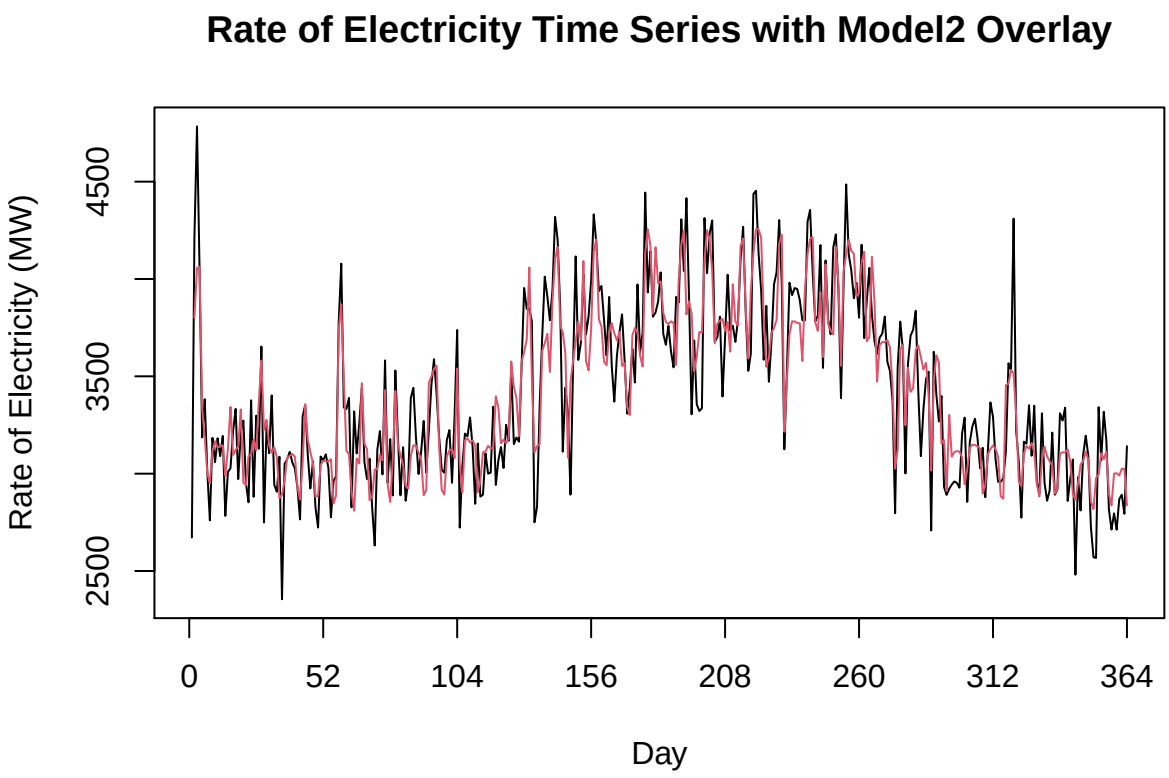
Despite losing significance in some of my coefficients, the new fit appears much improved. Naturally the explained variance in MW from the explanatory variables has increased due to a higher model complexity, and it has seen a jump to about 79%. The autoregressive coefficient has dropped to about 0.15, so while still mean reverting, it has changed from the base model by quite a bit.

The temperature ranges away from the mean temperature are positive as expected, and the extreme temperature values have a large coefficient associated with them; this is also what was expected. The current temperature value having a negative temperature is concerning. The majority of the temperatures shown in the previous temperature vs MW plot show that besides the small proportion of temperature values below 60, I regression coefficient slightly above 0 is what I would expect. There is the large proportion of temperature values that are relativley constant and then with a positive slope for the higher temperature values. The negative *cur.temp* coefficient (although not signficant at the 95% level) indicates that the small temperature values that have high leverage and a steep negative slope are affecting this coefficient.

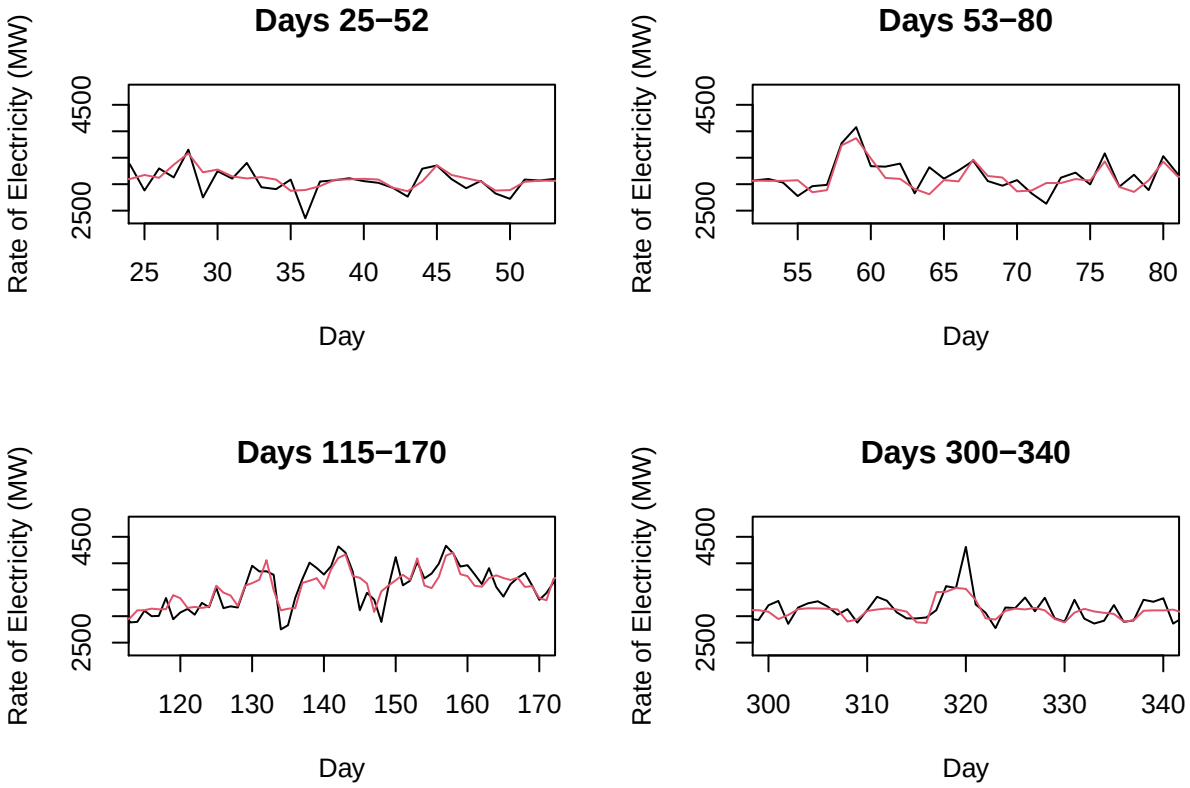
The residual plots show massive improvement. There are no longer the residuals that are as heavily underfit as before, and the histogram has lost the right side tail that both of my previous models had.

Let's see if these improvements are confirmed in the time series plot.

```
plot(elec$MW, type='l', xaxt='n', xlab='Day', ylab = 'Rate of Electricity (MW)',
     main='Rate of Electricity Time Series with Model2 Overlay')
axis(1, at=(0:7)*52, labels=((0:7)*52))
#adding line of fitted values
lines((time.l1), model2$fitted.values, col=2)
```




```
par(mfrow=c(2,2))
plot(elec$MW, type='l', xlab='Day', ylab = 'Rate of Electricity (MW)',
     main='Days 25-52', xlim=c(25,52))
lines((time.l1), model2$fitted.values, col=2)

plot(elec$MW, type='l', xlab='Day', ylab = 'Rate of Electricity (MW)',
     main='Days 53-80', xlim=c(53,80))
lines((time.l1), model2$fitted.values, col=2)

plot(elec$MW, type='l', xlab='Day', ylab = 'Rate of Electricity (MW)',
     main='Days 115-170', xlim=c(115,170))
lines((time.l1), model2$fitted.values, col=2)

plot(elec$MW, type='l', xlab='Day', ylab = 'Rate of Electricity (MW)',
     main='Days 300-340', xlim=c(300,340))
lines((time.l1), model2$fitted.values, col=2)
```

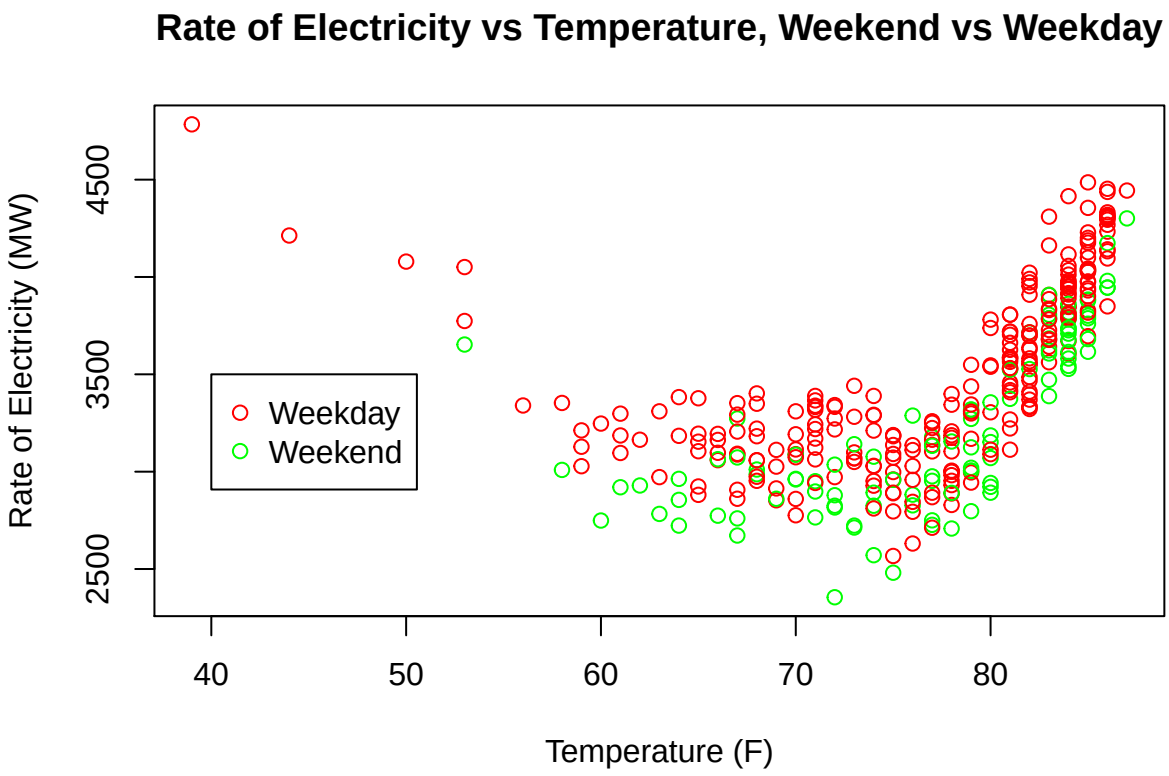


I think that the time series shows improvement. While far from acceptable, more of the peaks are being captured by my model as well as valleys in the time series for MW.

For a final model before BIC stepwise selection, I want to use interactions. I postulate that the peak MW values that aren't captured as accurately in my model might be due to an interaction of the day of the week with extreme temperature values. For now, I am going to fit one more model with the appropriate interaction.

I would like to see the difference in weekend vs weekday MW values against temperatures to test my postulation.

```
cols <- rep('red', nrow(elec))
cols[elec$DOW=='Sat'] <- 'green'
cols[elec$DOW=='Sun'] <- 'green'
plot(elec$temp, elec$MW, xlab='Temperature (F)', ylab = 'Rate of Electricity (MW)',
     main='Rate of Electricity vs Temperature, Weekend vs Weekday', col=cols, pch=21)
legend(40, 3500, legend=c("Weekday", "Weekend"), col=c("red", "green"), pch=21)
```



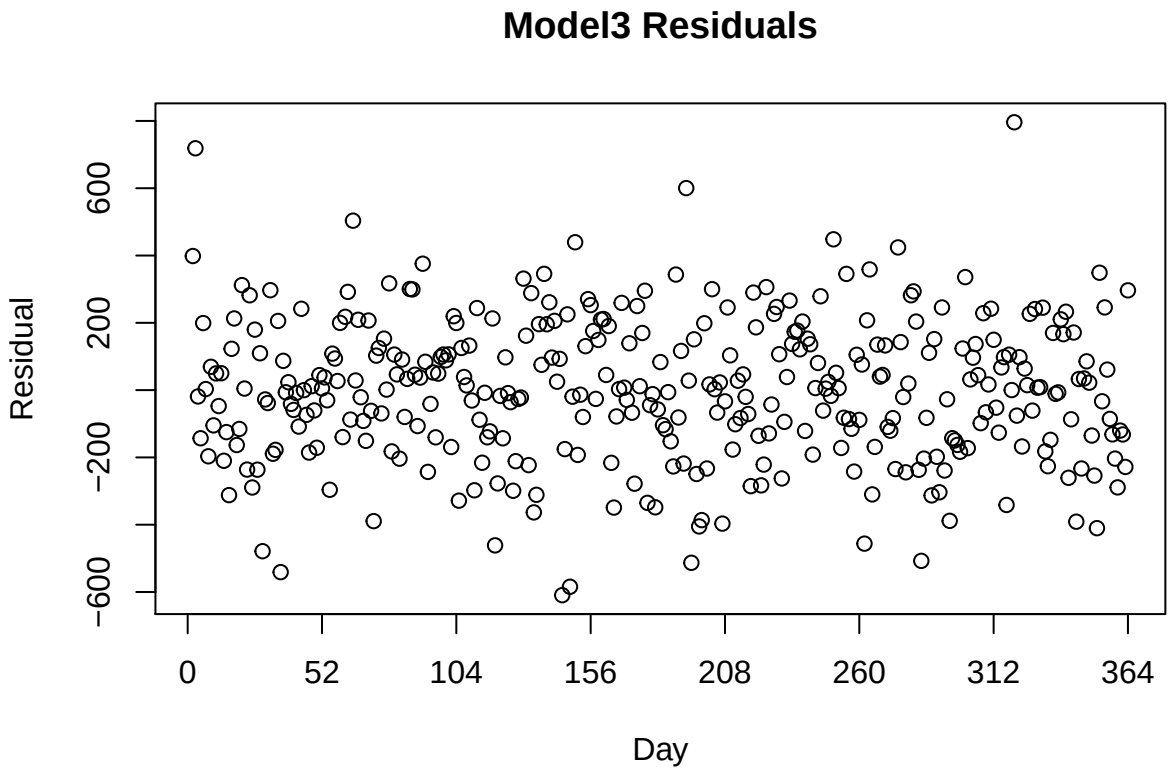
At the extreme temperature values, nearly all are the weekdays. I think an interaction may provide useful. Now to fitting a final autoregressive model before BIC stepwise selection.

```
model3 <- lm(MW~AR1 + cos300 + sin300 + weekend+ cur.temp + temp.range + temp.extreme +
               temp.extreme:weekend, data=train)
summary(model3)
```

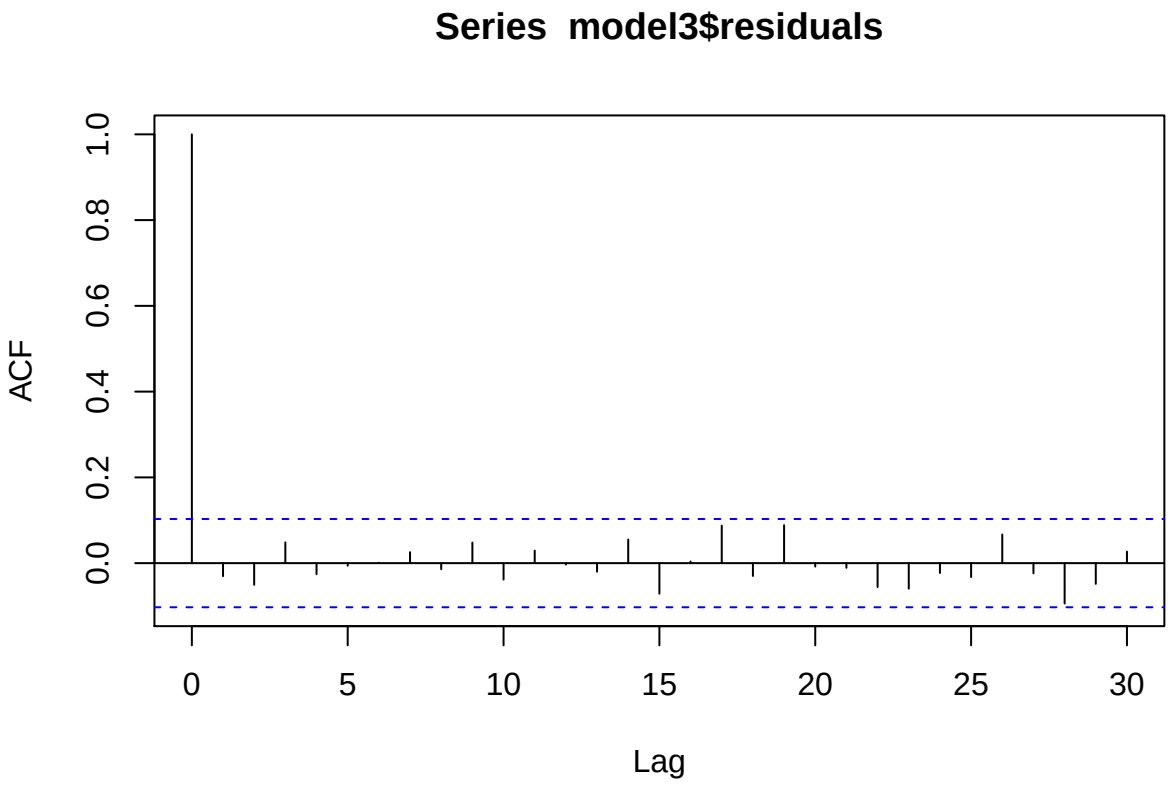
```
##
## Call:
## lm(formula = MW ~ AR1 + cos300 + sin300 + weekend + cur.temp +
##     temp.range + temp.extreme + temp.extreme:weekend, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -609.40 -139.88    3.13  138.24  795.99
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  3142.50844  270.93468  11.599  < 2e-16 ***
## AR1                             0.14644    0.03894   3.761 0.000198 ***
## cos300                        -60.83512   20.04349  -3.035 0.002583 **
## sin300                       -116.07934   29.02438  -3.999 7.74e-05 ***
## weekendTRUE                  -180.55605   26.80171  -6.737 6.60e-11 ***
## cur.temp                       -5.47580    3.15767  -1.734 0.083769 .
## temp.rangetemp<=60            175.59774   83.31550   2.108 0.035768 *
```

```
## temp.rangetemp>=80            411.55057   45.90190   8.966  < 2e-16 ***
## temp.extremeTRUE              411.71544   42.03970   9.793  < 2e-16 ***
## weekendTRUE:temp.extremeTRUE  -58.74610   71.61741  -0.820 0.412612
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 212.7 on 353 degrees of freedom
## Multiple R-squared:  0.7862, Adjusted R-squared:  0.7807
## F-statistic: 144.2 on 9 and 353 DF,  p-value: < 2.2e-16
```

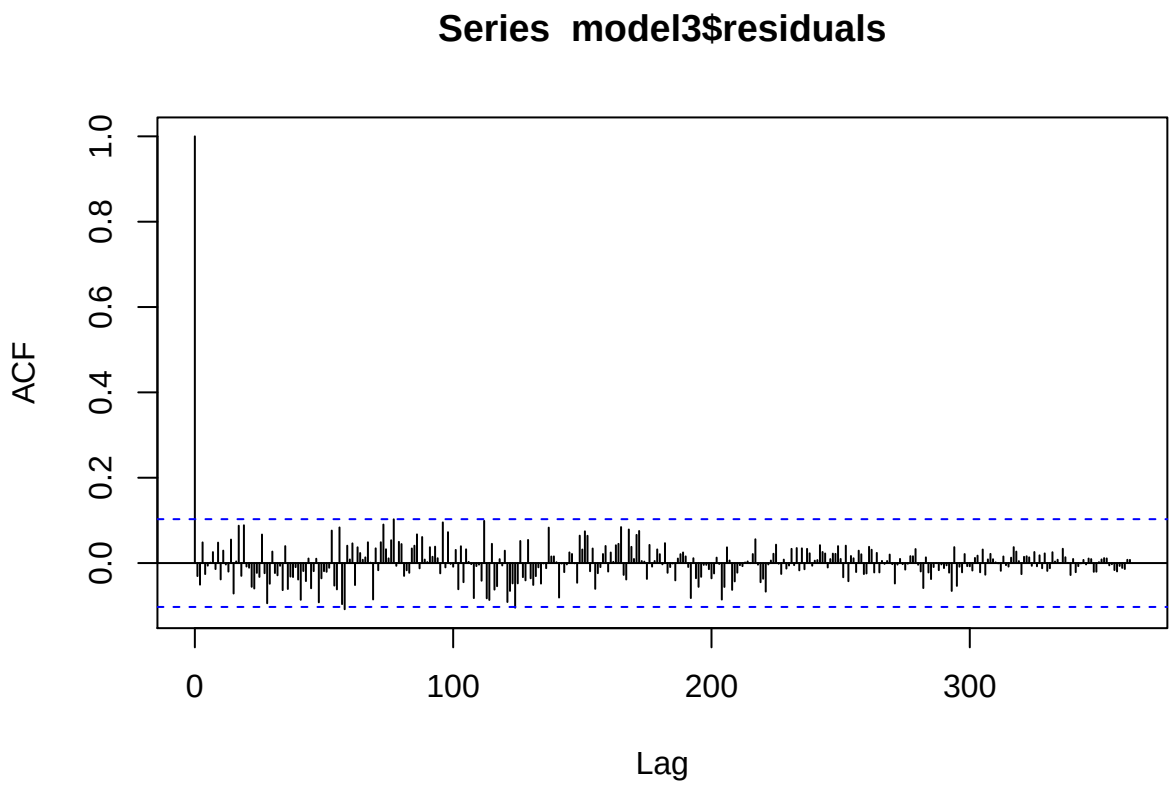
```
plot(time.l1, model3$residuals, xaxt='n', xlab='Day', ylab = 'Residual', main='Model3 Residuals')
axis(1, at=(0:7)*52, labels=((0:7)*52))
```



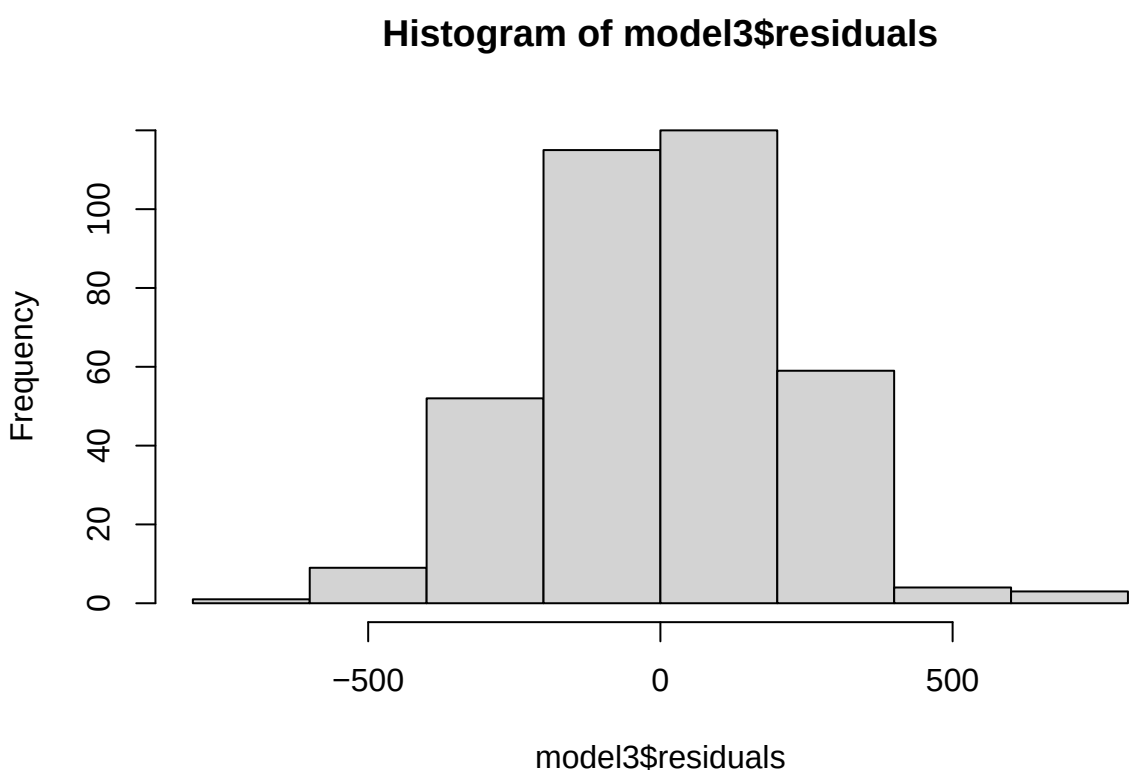

```
acf(model3$residuals, lag.max=30)
```




```
acf(model3$residuals, lag.max=365)
```

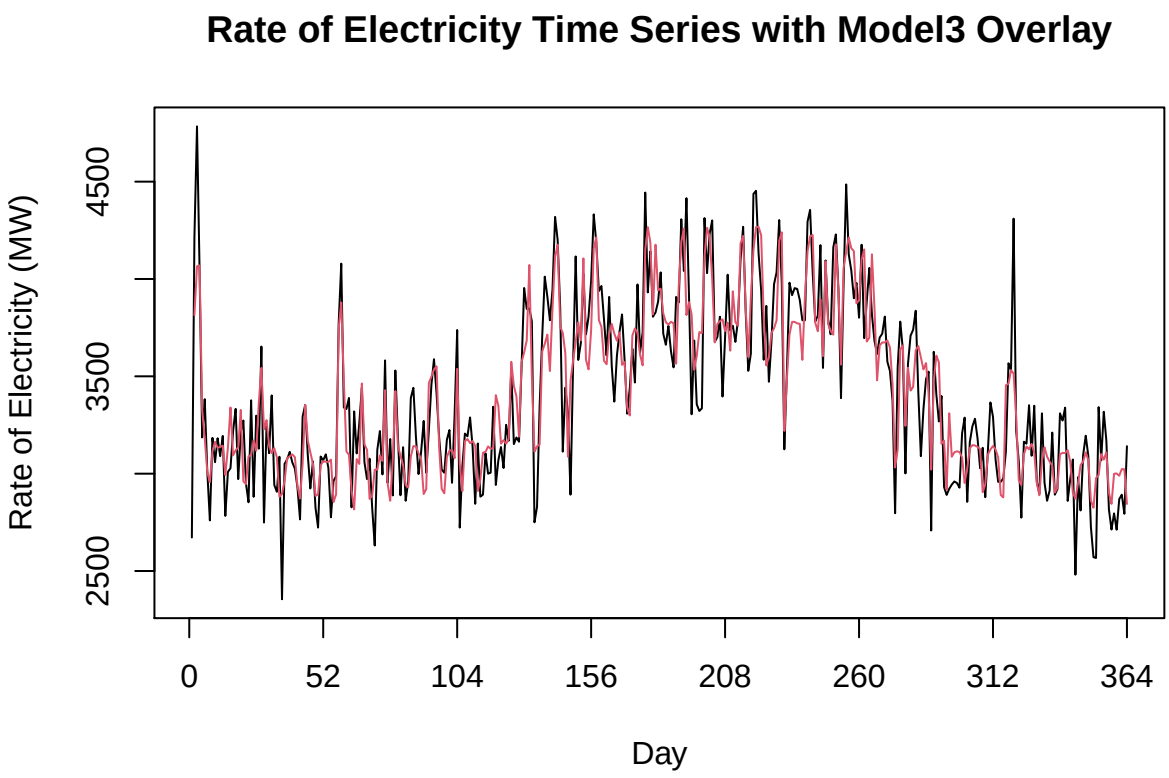



```
hist(model3$residuals)
```

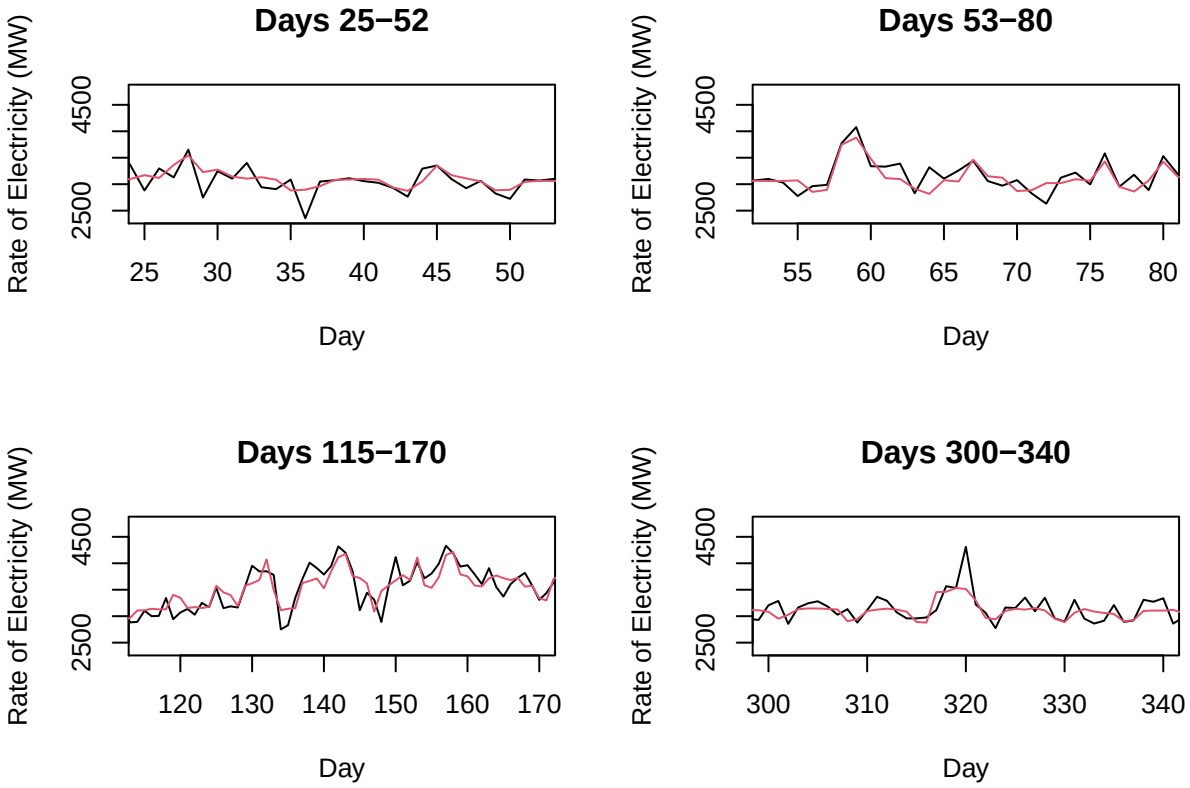


This final model shows little to no improvement. The residual diagnostics appear almost identical, and although my interaction has my expected negative coefficient for an interaction of weekends and extreme temperatures (I concluded the extreme temperatures with high MW values had almost exclusively weekdays), it is insignificant and other terms are now insignficicant. Let's see if the actual fit has improved.

```
plot(elec$MW, type='l', xaxt='n', xlab='Day', ylab = 'Rate of Electricity (MW)',
     main='Rate of Electricity Time Series with Model3 Overlay')
axis(1, at=(0:7)*52, labels=((0:7)*52))
#adding line of fitted values
lines((time.l1), model3$fitted.values, col=2)
```




```
par(mfrow=c(2,2))
plot(elec$MW, type='l', xlab='Day', ylab = 'Rate of Electricity (MW)',
     main='Days 25-52', xlim=c(25,52))
lines((time.l1), model3$fitted.values, col=2)

plot(elec$MW, type='l', xlab='Day', ylab = 'Rate of Electricity (MW)',
     main='Days 53-80', xlim=c(53,80))
lines((time.l1), model3$fitted.values, col=2)

plot(elec$MW, type='l', xlab='Day', ylab = 'Rate of Electricity (MW)',
     main='Days 115-170', xlim=c(115,170))
lines((time.l1), model3$fitted.values, col=2)

plot(elec$MW, type='l', xlab='Day', ylab = 'Rate of Electricity (MW)',
     main='Days 300-340', xlim=c(300,340))
lines((time.l1), model3$fitted.values, col=2)
```



The fit appears to be about the same. I don't think an interaction will be valuable in my model.

BIC Stepwise Model Selection

Now I will do BIC stepwise model selection. Before running the step function I will add one more dummy variable for just Sundays. Sundays appear to have even more local minima for their MW values associated with it. For this reason, I think that the extra dummy variable will be valuable.

For the stepwise selection to be considered successful, it must at the minumum improve the explained variance (R^2) and have less than 3 terms that are not statistically significant.

```
sunday <- as.factor(elec$DOW=='Sun')
sunday <- sunday[time.11]
train$sunday <- sunday
null <- lm(MW~1, data=train)
full <- lm(MW~ ., data=train)
forward <- step(object = null, scope = formula(full), direction='forward',
                k=log(nrow(train)), trace=F)
summary(forward)
```

```
##
## Call:
## lm(formula = MW ~ temp.range + temp.extreme + sunday + AR1 +
##     weekend + sin300 + cos300, data = train)
##
## Residuals:
```

```
##     Min      1Q  Median      3Q     Max
## -595.62 -145.38   -3.71  137.92  820.17
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)         2772.34638  128.66493  21.547  < 2e-16 ***
## temp.rangetemp<=60   283.79293   60.75823   4.671 4.27e-06 ***
## temp.rangetemp>=80   373.45164   38.85191   9.612  < 2e-16 ***
## temp.extremeTRUE     390.19348   36.28258  10.754  < 2e-16 ***
## sundayTRUE          -162.30519   41.43996  -3.917 0.000108 ***
## AR1                    0.13630    0.03824   3.564 0.000415 ***
## weekendTRUE         -110.26976   31.86098  -3.461 0.000604 ***
## sin300              -109.16176   27.87032  -3.917 0.000108 ***
## cos300               -50.00152   18.32024  -2.729 0.006664 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 209.1 on 354 degrees of freedom
## Multiple R-squared:  0.7929, Adjusted R-squared:  0.7882
## F-statistic: 169.4 on 8 and 354 DF,  p-value: < 2.2e-16
```

Model Comparison

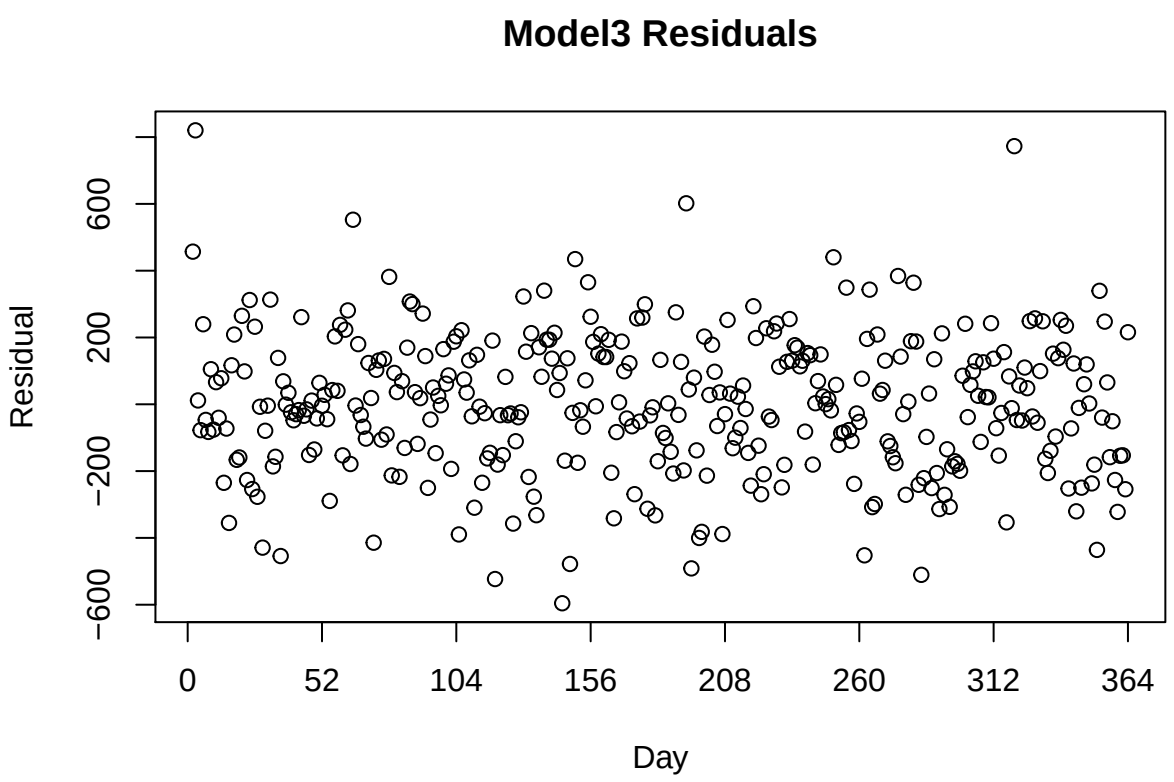
Now let's use BIC to compare all of my models.

```
BIC <- c(baseline=extractAIC(base.model, k=log(nrow(train)))[2],
         mod1=extractAIC(model1, k=log(nrow(train)))[2],
         mod2=extractAIC(model2, k=log(nrow(train)))[2],
         mod3=extractAIC(model3, k=log(nrow(train)))[2],
         bic.step=extractAIC(forward, k=log(nrow(train)))[2])
eBIC <- exp(-0.5*(BIC-min(BIC)))
probs <- eBIC/sum(eBIC)
round(probs, 5)
```

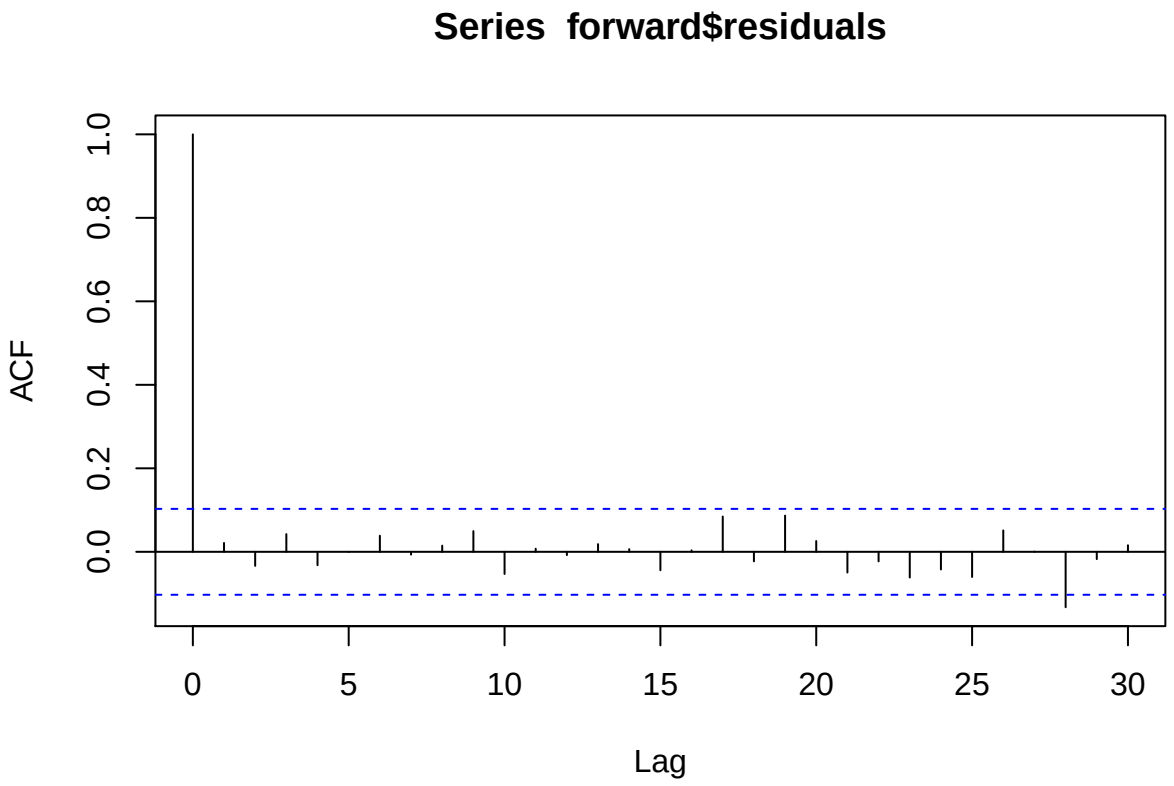
```
## baseline     mod1     mod2     mod3 bic.step
##  0.00000  0.00000  0.00220  0.00016  0.99764
```

With probability 0.99764, my forward stepwise selected model is the correct model. Let's check the model assumptions, then view the time series.

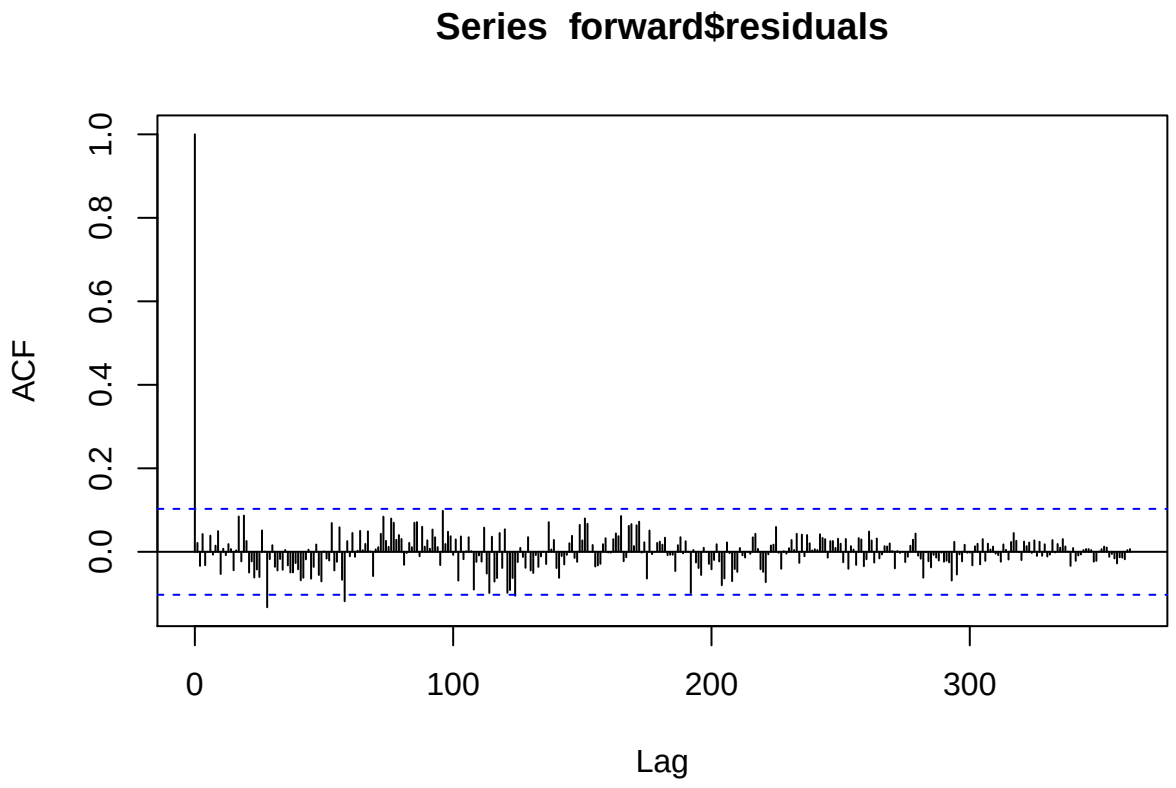
```
plot(time.l1, forward$residuals, xaxt='n', xlab='Day', ylab = 'Residual',
     main='Model3 Residuals')
axis(1, at=(0:7)*52, labels=((0:7)*52))
```



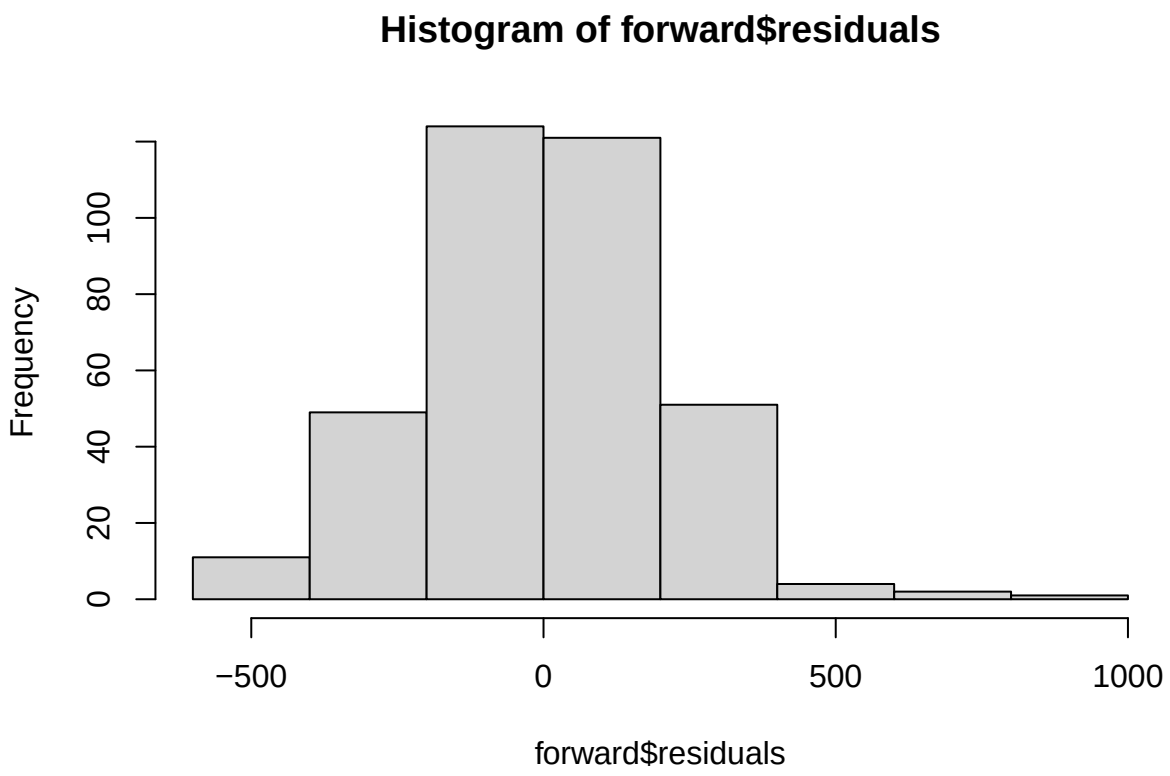

```
acf(forward$residuals, lag.max=30)
```



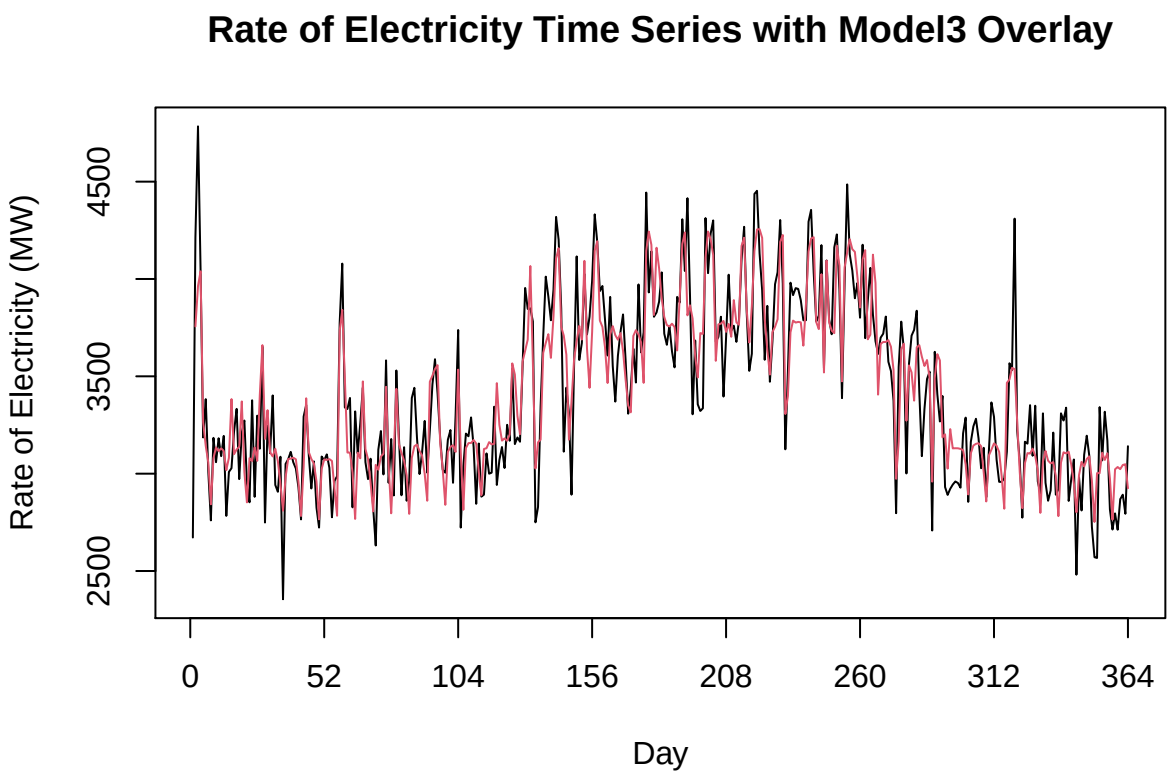

```
acf(forward$residuals, lag.max=365)
```



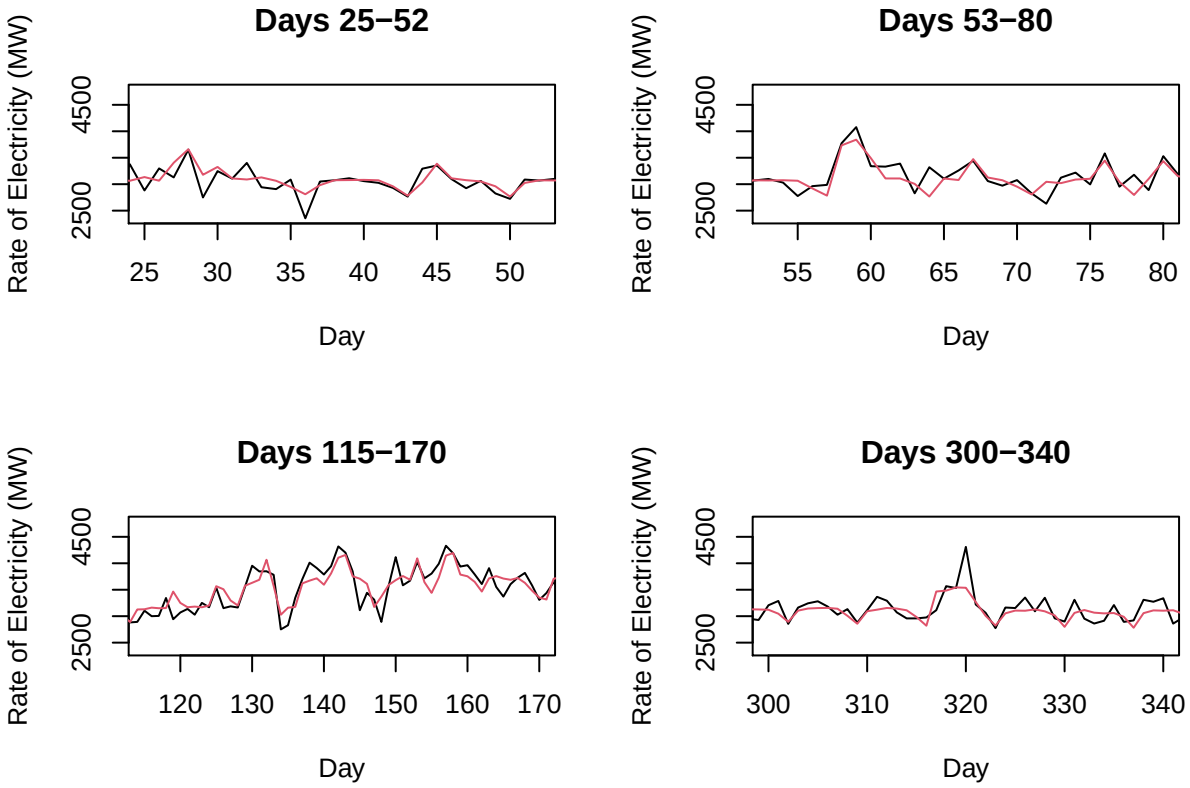

```
hist(forward$residuals)
```




```
plot(elec$MW, type='l', xaxt='n', xlab='Day', ylab = 'Rate of Electricity (MW)',
     main='Rate of Electricity Time Series with Model3 Overlay')
axis(1, at=(0:7)*52, labels=((0:7)*52))
#adding line of fitted values
lines((time.l1), forward$fitted.values, col=2)
```




```
par(mfrow=c(2,2))
plot(elec$MW, type='l', xlab='Day', ylab = 'Rate of Electricity (MW)',
     main='Days 25-52', xlim=c(25,52))
lines((time.l1), forward$fitted.values, col=2)

plot(elec$MW, type='l', xlab='Day', ylab = 'Rate of Electricity (MW)',
     main='Days 53-80', xlim=c(53,80))
lines((time.l1), forward$fitted.values, col=2)

plot(elec$MW, type='l', xlab='Day', ylab = 'Rate of Electricity (MW)',
     main='Days 115-170', xlim=c(115,170))
lines((time.l1), forward$fitted.values, col=2)

plot(elec$MW, type='l', xlab='Day', ylab = 'Rate of Electricity (MW)',
     main='Days 300-340', xlim=c(300,340))
lines((time.l1), forward$fitted.values, col=2)
```



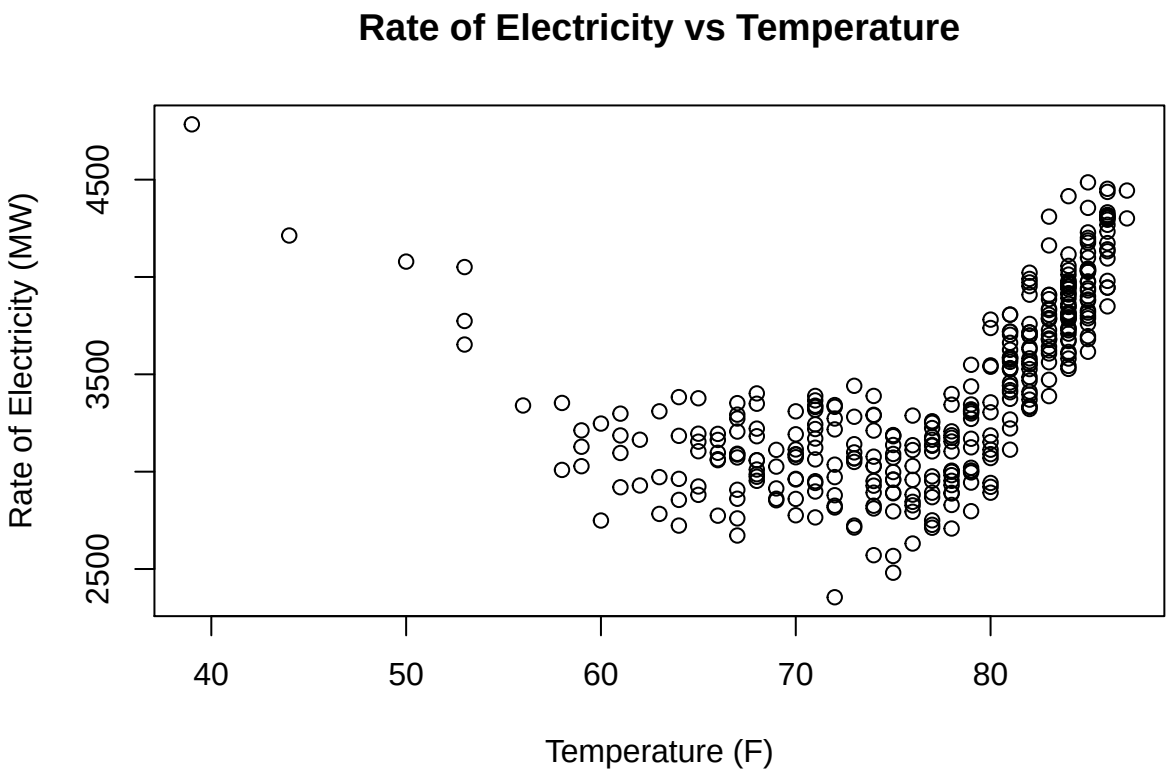
If not for some residual outliers the autoregressive model assumptions for residuals would be met. With my final model, I am dissapointed. While my model has a decent fit on the time series of MW values, It clearly is lacking severely and I will not feel confident in my forecasting results. More time could be invested to improve this model through the creation of more features (at which point I risk overfitting).

This previous model had a lot invested into it in terms of time and exploration. I hope this next model, which I expect to be much simpler, is able to be a better fit for the data.

Analysis: Multiple Linear Regression Model for MW values

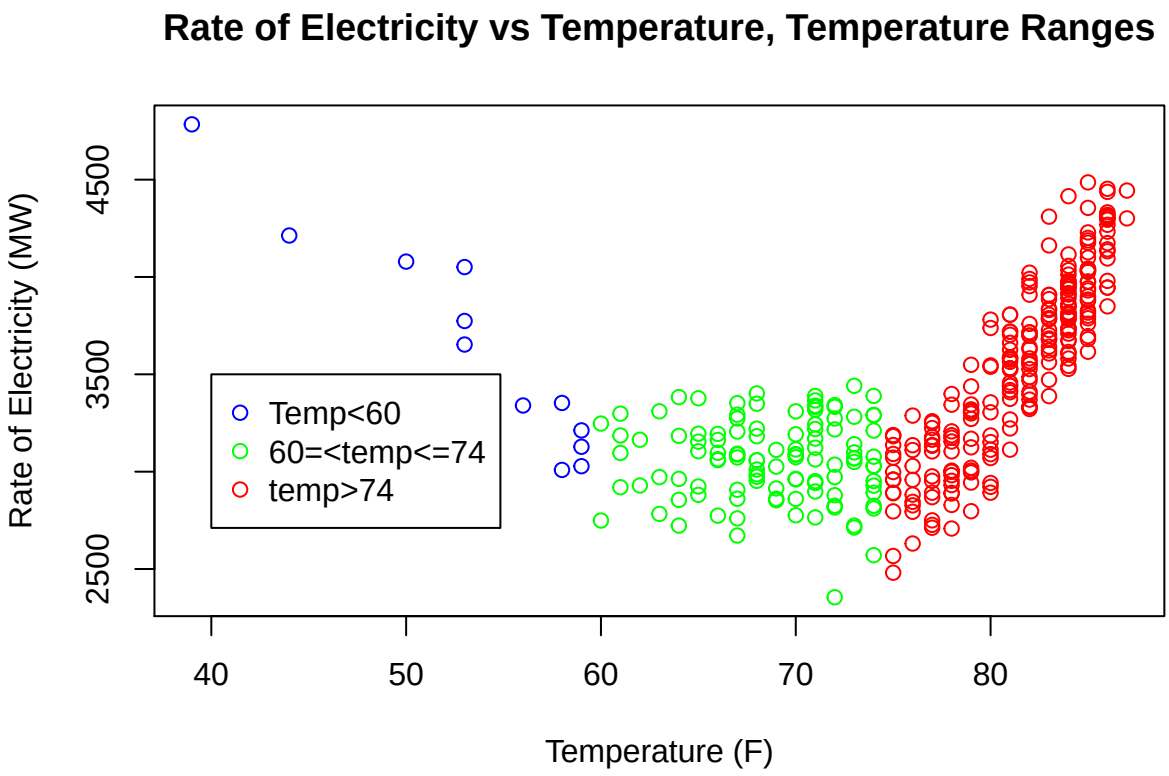
Returning back to a simpler, less messy plot: MW vs *temp*.

```
plot(elec$temp, elec$MW, xlab='Temperature (F)', ylab = 'Rate of Electricity (MW)',
     main='Rate of Electricity vs Temperature', pch=21)
```



I should be able to have a strong model for this data if I divide the temperatures into 3 ranges for 3 different lines. A segmentation with the following temperature ranges should be suitable.

```
# colors to be filled
cols <- rep('green', nrow(elec))
cols[elec$temp<60] <- 'blue'
cols[elec$temp>74] <- 'red'

plot(elec$temp, elec$MW, xlab='Temperature (F)', ylab = 'Rate of Electricity (MW)',
     main='Rate of Electricity vs Temperature, Temperature Ranges', col=cols, pch=21)
legend(40, 3500, legend=c("Temp<60", "60=<temp<=74", 'temp>74'),
       col=c("blue", "green", 'red'), pch=21)
```



In order to achieve these 3 seperate fits, all that is needed is a dummy variable that acts as indicator of the temperature ranges. These will be labeled 'low', 'med', and 'high'.

```
# these are the temperature values in the respective ranges
low <- elec$temp[elec$temp < 60]
med <- elec$temp[elec$temp >=60 & elec$temp <=74]
high <- elec$temp[elec$temp>74]

# and this dummy variable is the indicator for what range a temperature is in
temper.range <- rep('med',nrow(elec))
temper.range[elec$temp < 60] <- 'low'
temper.range[elec$temp >74] <- 'high'
temper.range <- as.factor(temper.range)
low <- elec$temp[elec$temp < 60]
med <- elec$temp[elec$temp >=60 & elec$temp <=74]
high <- elec$temp[elec$temp>74]
```

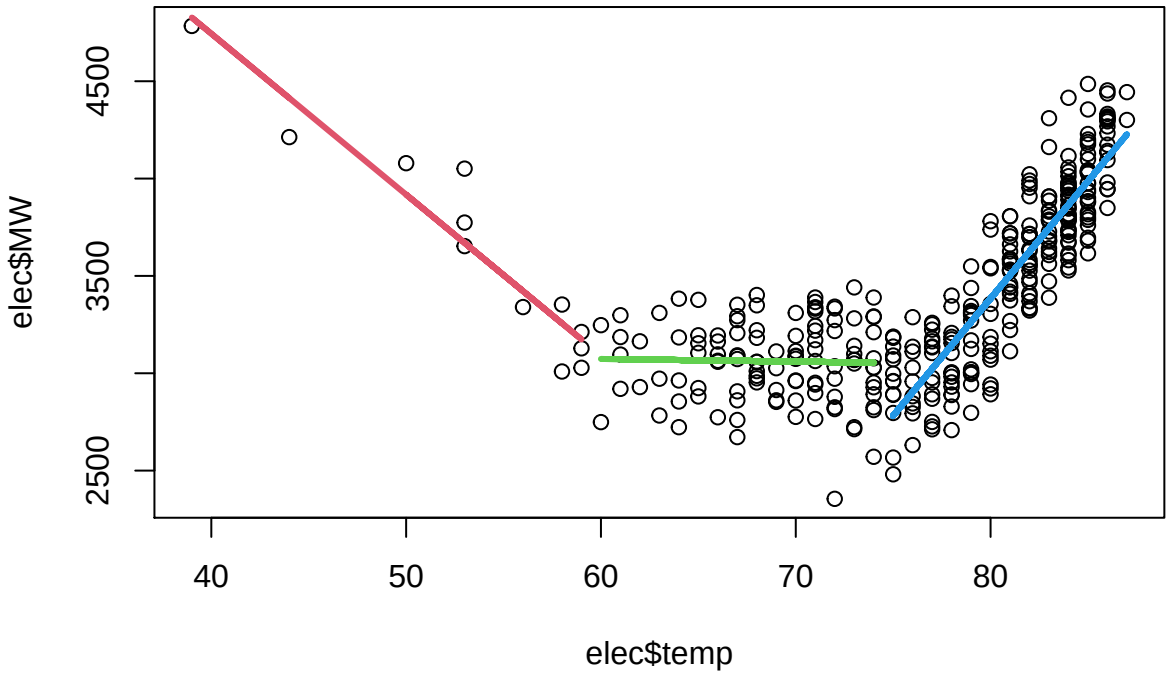
Now, a multiple linear regression model can be fit to the data, using an interaction term to get the 3 seperate lines needed for our data.

```
model.MLR <- lm(MW~ temp+ temper.range + temp:temper.range, data=elec)
summary(model.MLR)
```

```
##
## Call:
## lm(formula = MW ~ temp + temper.range + temp:temper.range, data = elec)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -702.26 -144.12   12.52  143.04  564.58
##
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)          -6245.915    332.428  -18.79   <2e-16 ***
## temp                   120.378      4.080   29.51   <2e-16 ***
## temper.rangelow      14296.184    622.586   22.96   <2e-16 ***
## temper.rangemed       9400.421    490.920   19.15   <2e-16 ***
## temp:temper.rangelow  -203.028     10.606  -19.14   <2e-16 ***
## temp:temper.rangemed  -121.728      6.638  -18.34   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 207.9 on 358 degrees of freedom
## Multiple R-squared:  0.7943, Adjusted R-squared:  0.7915
## F-statistic: 276.6 on 5 and 358 DF,  p-value: < 2.2e-16
```

```
plot(elec$temp, elec$MW)
lines(elec$temp[elec$temp<60], model.MLR$fitted.values[elec$temp<60], col=2, lwd=3)
lines(elec$temp[elec$temp>=60 & elec$temp<=74],model.MLR$fitted.values[elec$temp>=60&elec$temp<=74],
      col=3, lwd=3)
lines(elec$temp[elec$temp>74],model.MLR$fitted.values[elec$temp>74], col=4, lwd=3)
```



All three temperature ranges have a regression line fit to their temperatures. Here is a better visualization of the data.

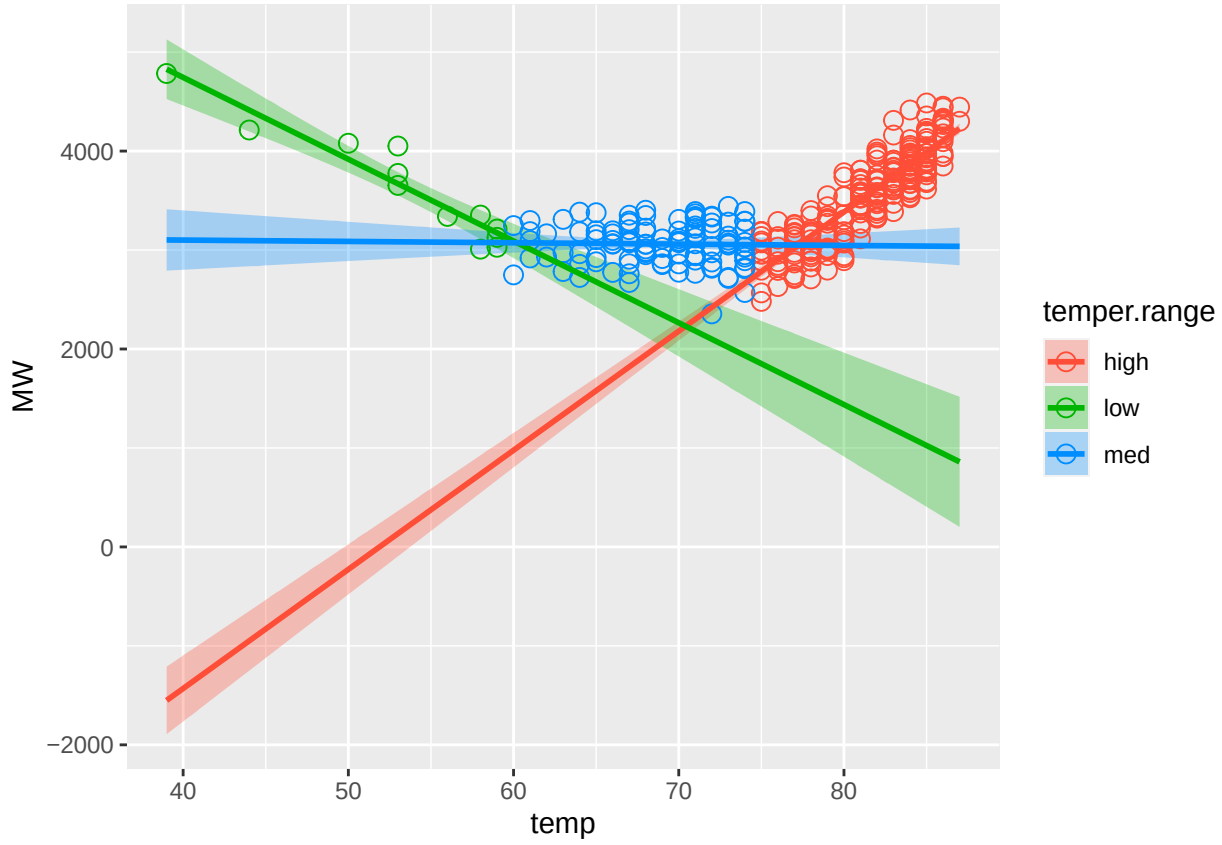
```
library(visreg);
```

```
## Warning: package 'visreg' was built under R version 4.0.3
```

```
library(ggplot2);
```

```
## Warning: package 'ggplot2' was built under R version 4.0.2
```

```
visreg(model.MLR, "temp", gg=T, by='temper.range', overlay=T, rug=F,
       type='conditional', points=list(pch=21, size=3))
```

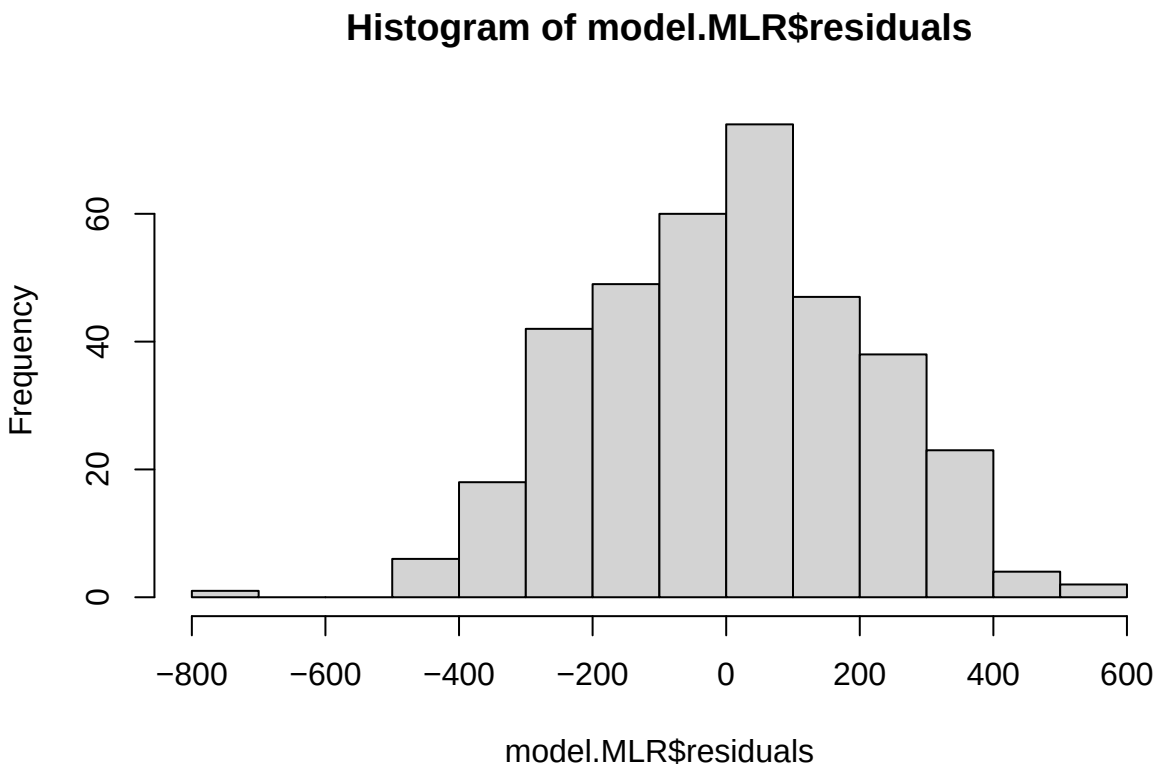


With the 95% confidence interval bands displayed, the three seperate fits for different temperature ranges are visualized above.

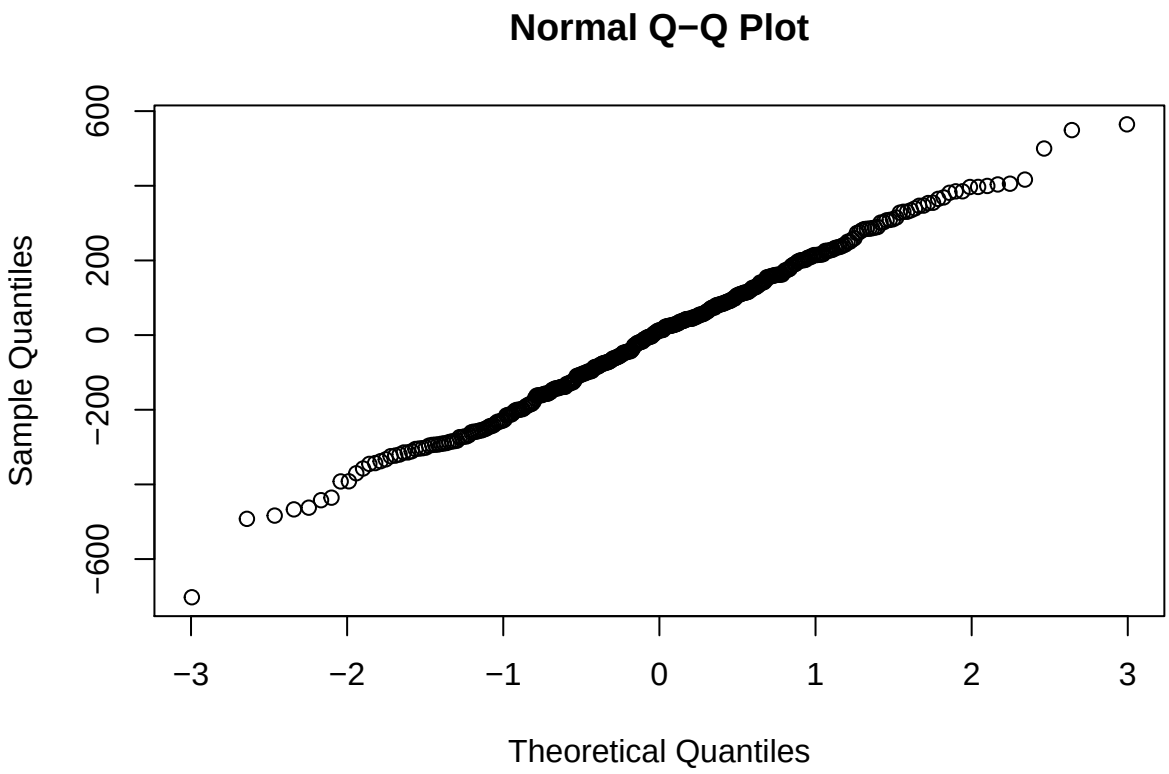
The summary of the fit is suprisingly good. All of the terms have are very statistically significant at the 95% level, and we have an explained variance for the MW values of about 79.43%. This already surpasses the 79.29% from the best autoregressive model.

Let's check the residuals for this model.

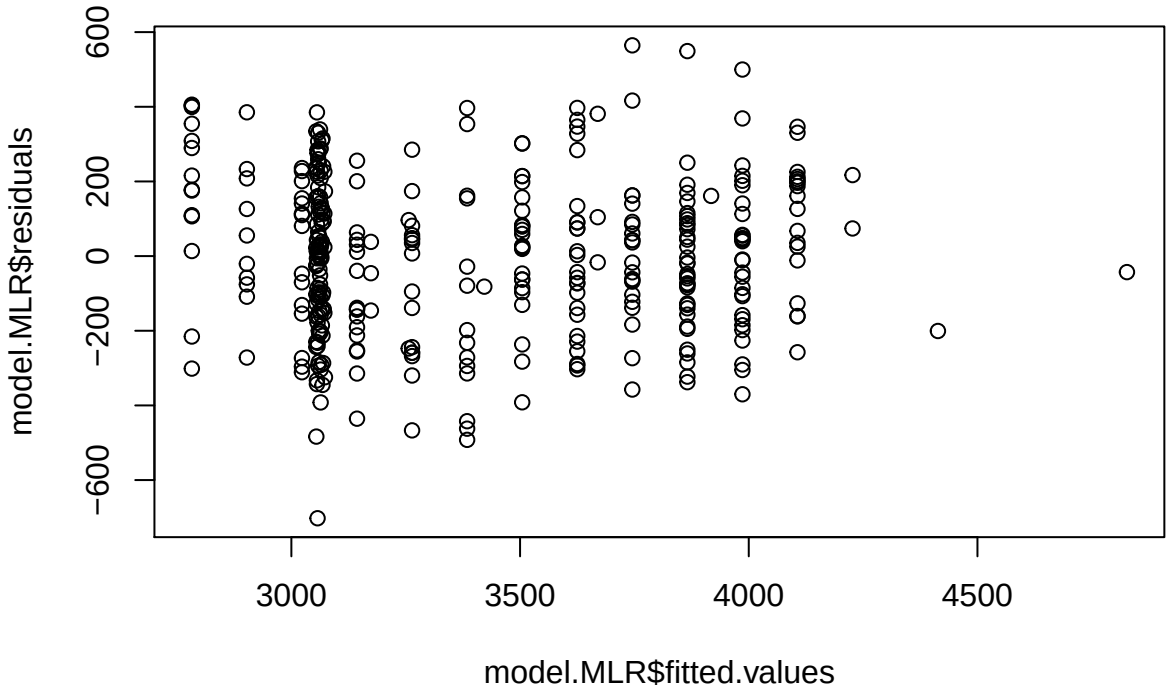
```
hist(model.MLR$residuals)
```




```
qqnorm(model.MLR$residuals)
```




```
plot(model.MLR$fitted.values, model.MLR$residuals)
```



Besides an outlier, the residual assumptions for a linear regression are met. Residuals are normally distrubted with mean 0 and a constant variance.

To improve this model, I will create a frame to remove this outlier, and try adding one of my previously successful dummy variables from the autoregressive model: weekends.

```
# training frame for the MLR models
# the outlier has a value of 2355 (computed in console)
train.MLR <- data.frame(MW=elec$MW[elec$MW > 2360], temp = elec$temp[elec$MW>2360],
                        temper.range=temper.range[elec$MW >2360])
#adding weekend variable
weekend <- rep(FALSE, nrow(elec))
weekend[elec$DOW == 'Sun'] <- TRUE
weekend[elec$DOW == 'Sat'] <- TRUE
weekend <- as.factor(weekend[elec$MW>2360])
train.MLR$weekend <- weekend
```

Now I can fit a new model with the weekend variable added in

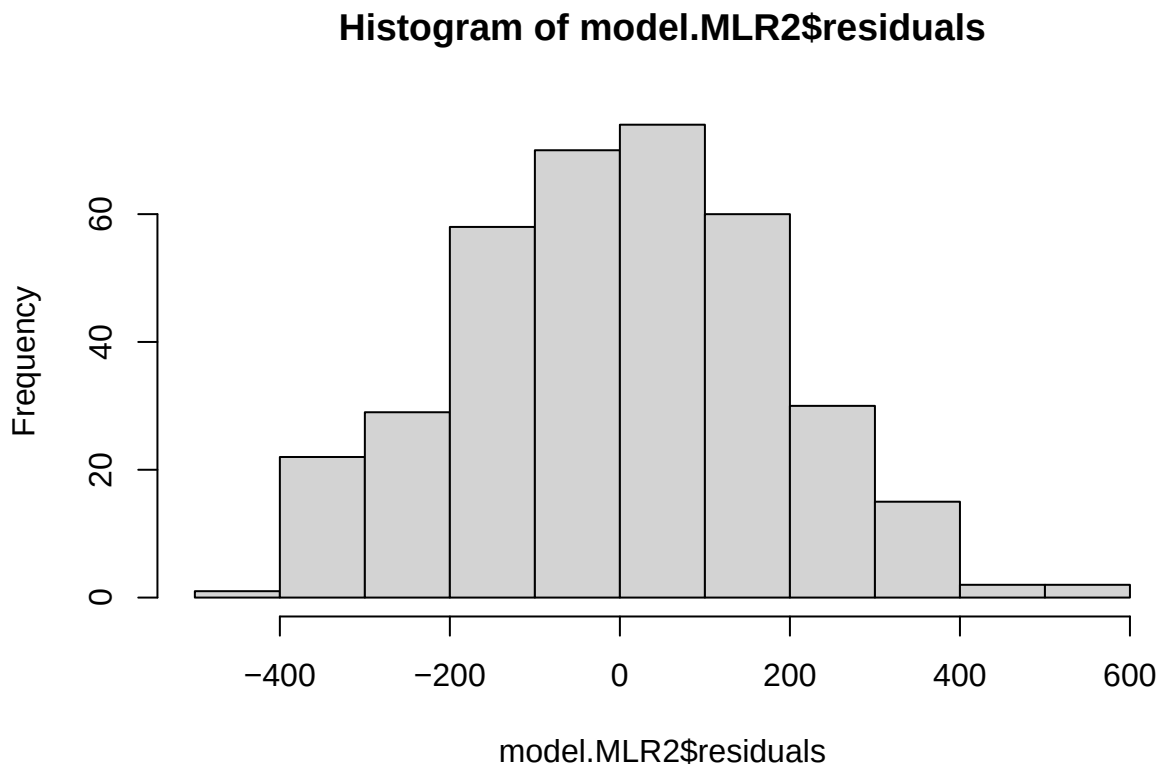
```
model.MLR2 <- lm(MW~temp+temper.range+weekend+temp:temper.range, data=train.MLR)
summary(model.MLR2)
```

```
##
## Call:
## lm(formula = MW ~ temp + temper.range + weekend + temp:temper.range,
##     data = train.MLR)
```

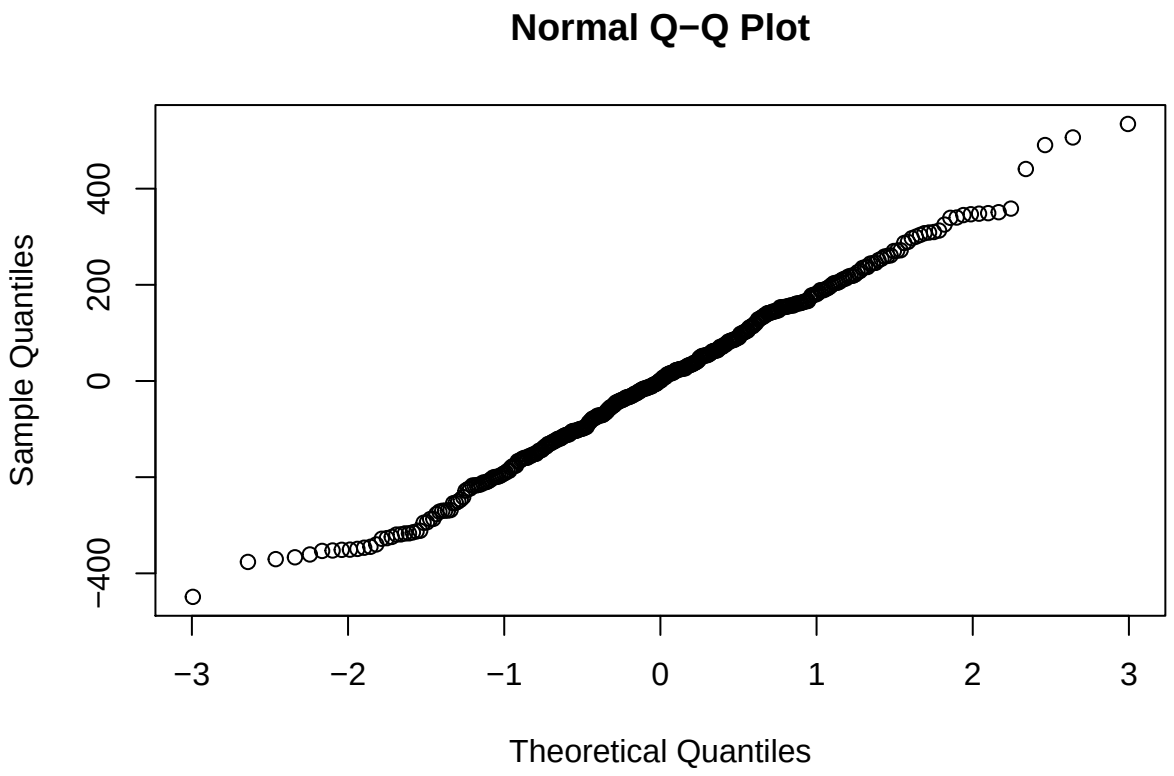
```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -448.89 -123.93    0.78  137.19  534.49
##
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)          -6224.508    292.415 -21.287   <2e-16 ***
## temp                   120.820      3.589  33.665   <2e-16 ***
## temper.rangelow      14207.992    547.709  25.941   <2e-16 ***
## temper.rangemed       9381.518    432.513  21.691   <2e-16 ***
## weekendTRUE           -204.279     21.329  -9.578   <2e-16 ***
## temp:temper.rangelow  -201.583      9.331 -21.604   <2e-16 ***
## temp:temper.rangemed  -121.215      5.850 -20.719   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 182.9 on 356 degrees of freedom
## Multiple R-squared:  0.8394, Adjusted R-squared:  0.8367
## F-statistic: 310.1 on 6 and 356 DF,  p-value: < 2.2e-16
```

Residuals:

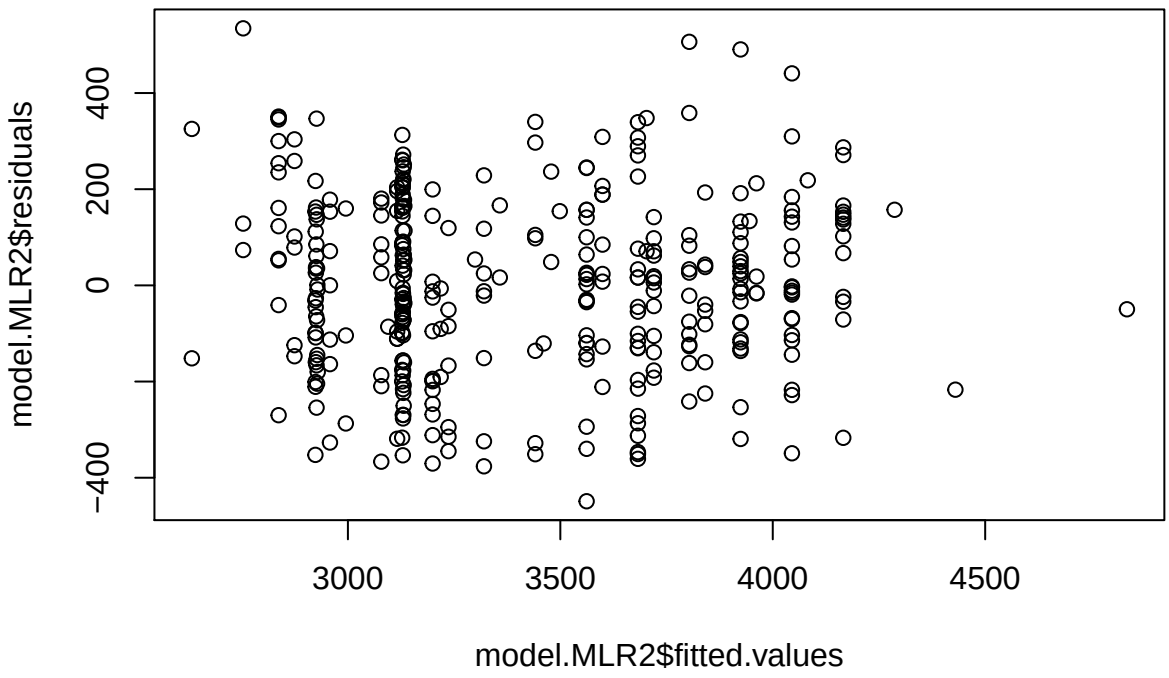
```
hist(model.MLR2$residuals)
```



```
qqnorm(model.MLR2$residuals)
```

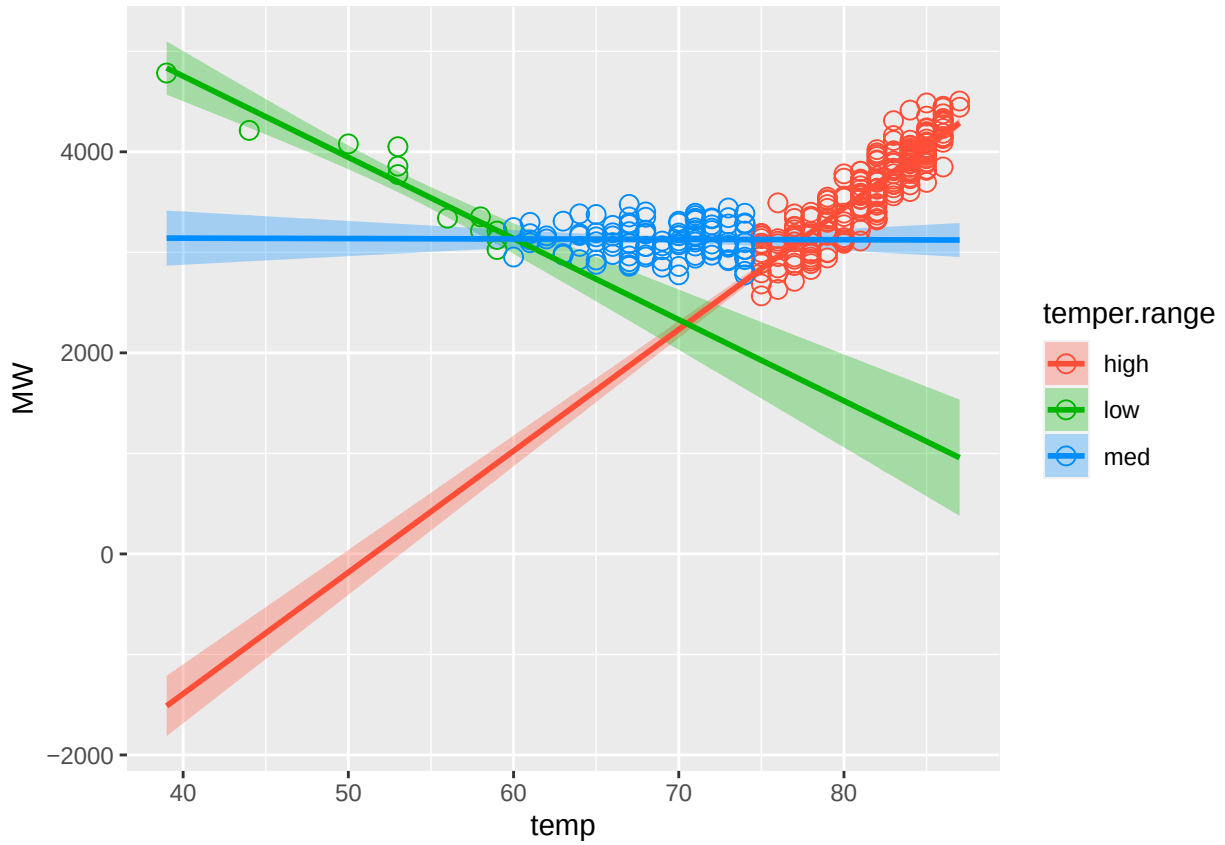



```
plot(model.MLR2$fitted.values, model.MLR2$residuals)
```



The summary output for this model shows an improved fit. The R^2 value indicating variance in MW explained by the explanatory variables has improved over 4 percentage points to 83.94%. Also, all of the regression coefficients retain their very high level of statistical significance. The distribution of the residuals seems to be improved, and the residual assumptions of normality are met.

```
visreg(model.MLR2, "temp", gg=T, by=c('temper.range'), overlay=T,
       rug=F, type='conditional', points=list(pch=21, size=3))
```



I will not compare these 2 models I have just created, then compare the best of the two to the autoregressive model.

```
BIC <- c(MLR1=extractAIC(model.MLR, k=log(nrow(elec)))[2],
         MLR2=extractAIC(model.MLR2, k=log(nrow(train.MLR)))[2])
eBIC <- exp(-0.5*(BIC-min(BIC)))
probs <- eBIC/sum(eBIC)
round(probs, 5)
```

```
## MLR1 MLR2
##    0    1
```

The second MLR model has a probability of 1 of being the correct model between the two models. Let's compare this model to the autoregressive model.

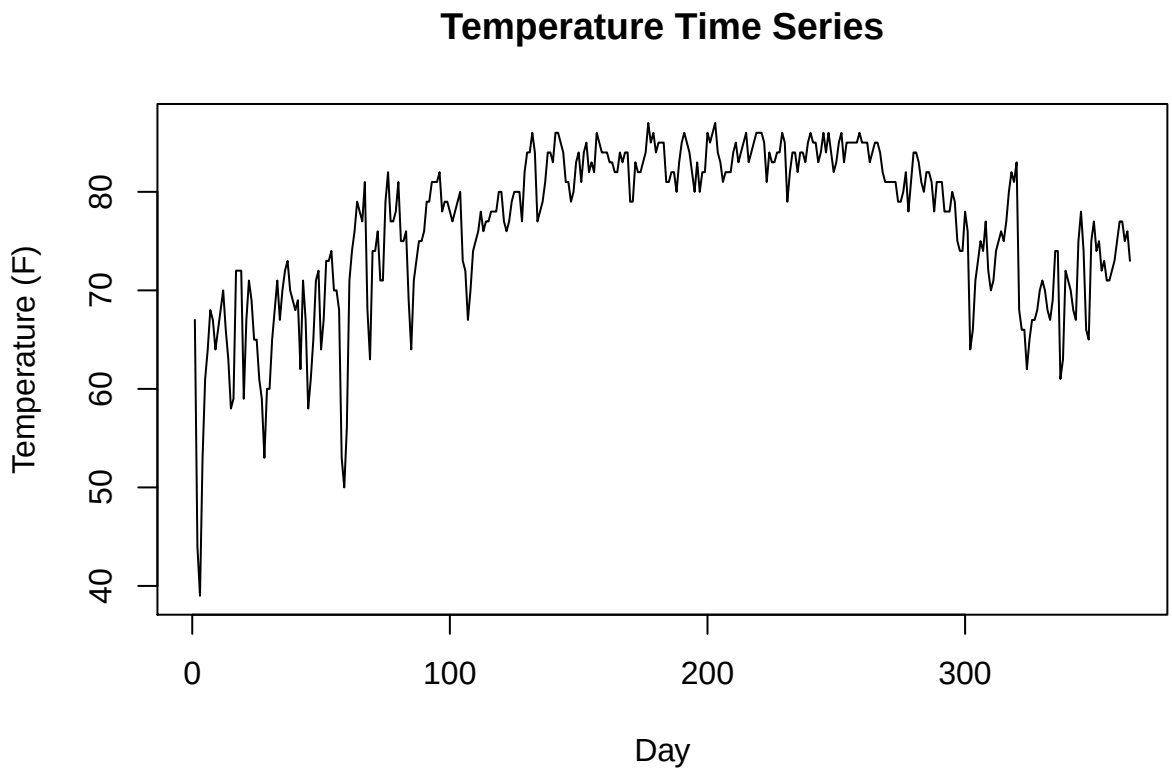
```
BIC <- c(forward=extractAIC(forward, k=log(nrow(train)))[2],
         MLR2=extractAIC(model.MLR2, k=log(nrow(train.MLR)))[2])
eBIC <- exp(-0.5*(BIC-min(BIC)))
probs <- eBIC/sum(eBIC)
round(probs, 5)
```

```
## forward    MLR2
##       0       1
```

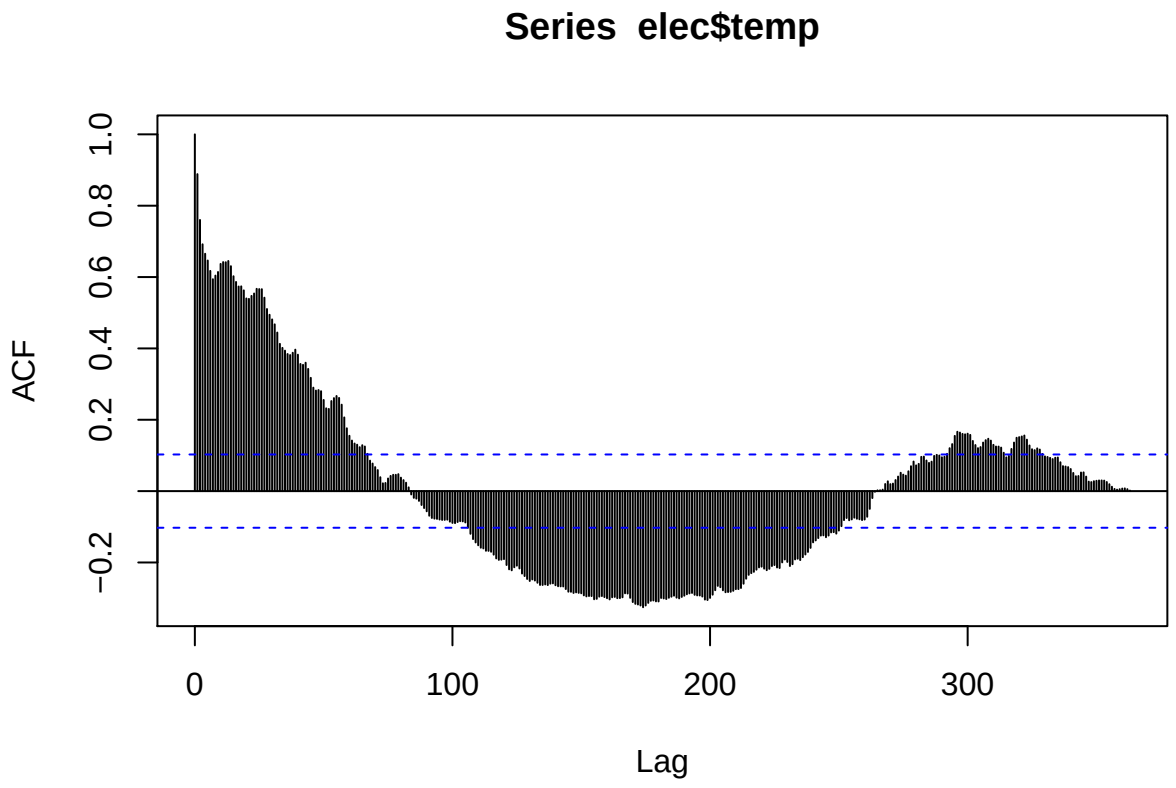
As I had anticipated, the multiple linear regression has a probality 1 of being the correct model when compared with the best autoregressive model I had selected. This result means that I will use this MLR model for prediction. But first comes one last model: an autoregressive model for temperature.

Temperature Autoregressive Model My frame from my *MW* autoregressive model has many of the important variables needed for this model. Before I start, I want to check the acf plot of the temperature time series in order to make sure I know the periodicity of the data.

```
plot(elec$temp, type='l', xlab='Day', ylab='Temperature (F)',
     main='Temperature Time Series')
```




```
acf(elec$temp, lag.max=365)
```



A a frequency of about 310 should be suitable for the time series. Now I will build a dataframe with variables for the *temp* autoregressive model.

```
train.temp = data.frame(cur.temp = cur.temp, prev.temp = prev.temp,
                        cos310 = cos(2*pi*time.l1/310), sin310 = sin(2*pi*time.l1/310))
```

I believe all of the added terms will be viable in the model, and for this reason I will start by fitting a model with all the terms.

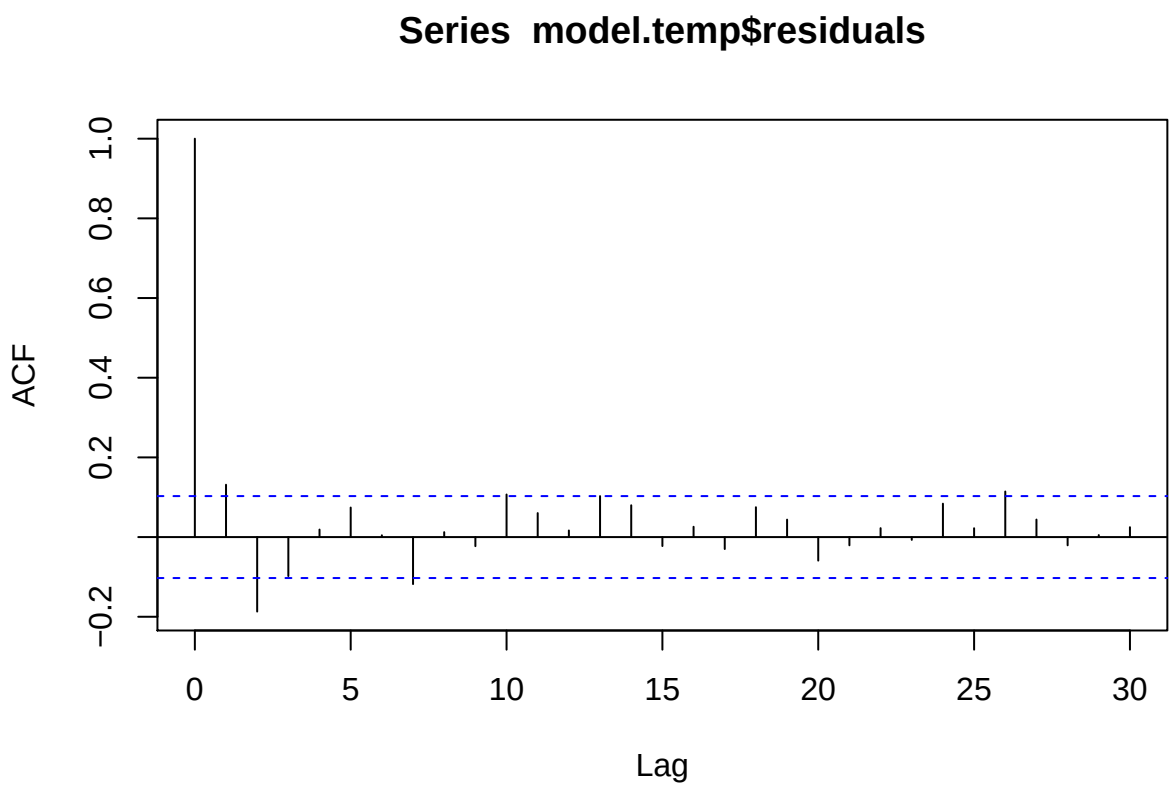
```
model.temp <- lm(cur.temp ~., data=train.temp)
summary(model.temp)
```

```
##
## Call:
## lm(formula = cur.temp ~ ., data = train.temp)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -24.2298  -1.2248   0.4688   1.6783  11.2995
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 19.78105    2.73626   7.229 2.95e-12 ***
## prev.temp    0.74606    0.03508  21.269  < 2e-16 ***
## cos310      -1.47937    0.32166  -4.599 5.89e-06 ***
## sin310      -1.45475    0.34700  -4.192 3.48e-05 ***
```

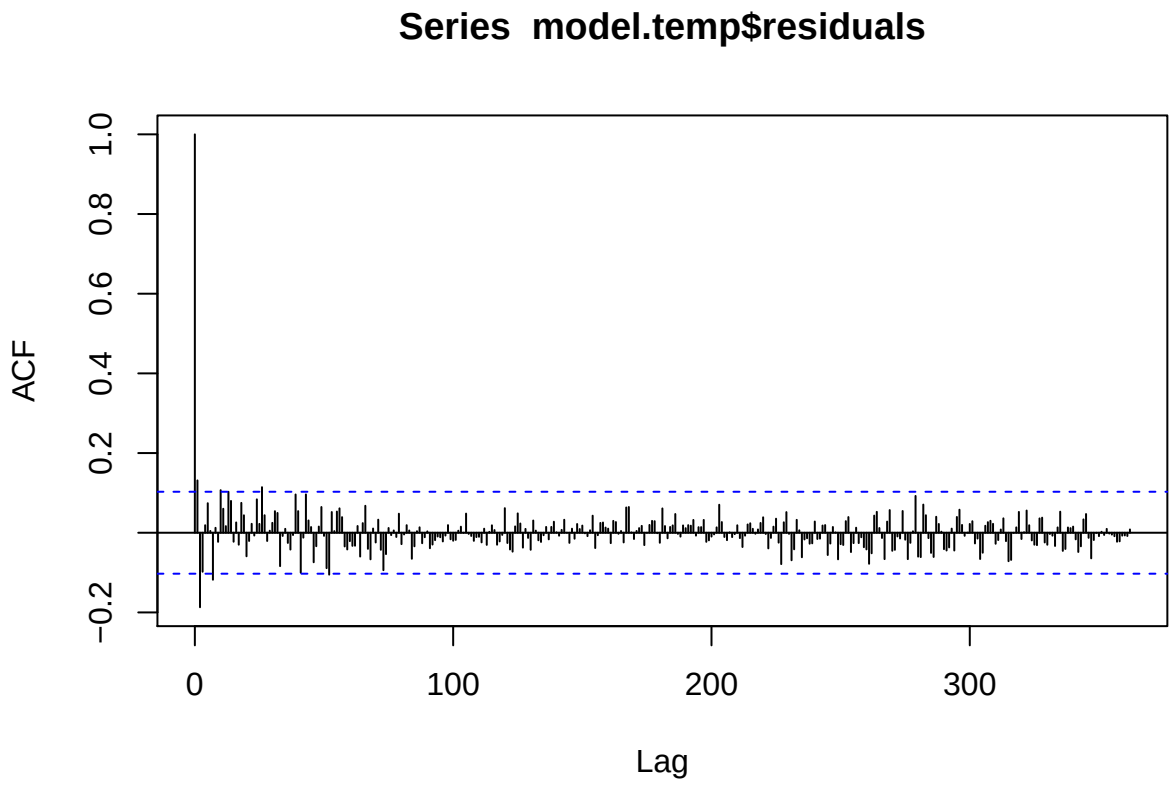
```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.51 on 359 degrees of freedom
## Multiple R-squared:  0.8088, Adjusted R-squared:  0.8072
## F-statistic: 506.2 on 3 and 359 DF,  p-value: < 2.2e-16
```

All variables are significant, with the previous temperature variable (the autoregressive term of lag=1) having an absolute value less than 1. This means the autoregressive model is mean reverting, which is preferable for my forecasting. There is also an R^2 value of 0.8088 that is decent (or at least not horrible, I do not know what good R^2 values in the best temperature autoregressive models are). Looking at the residuals.

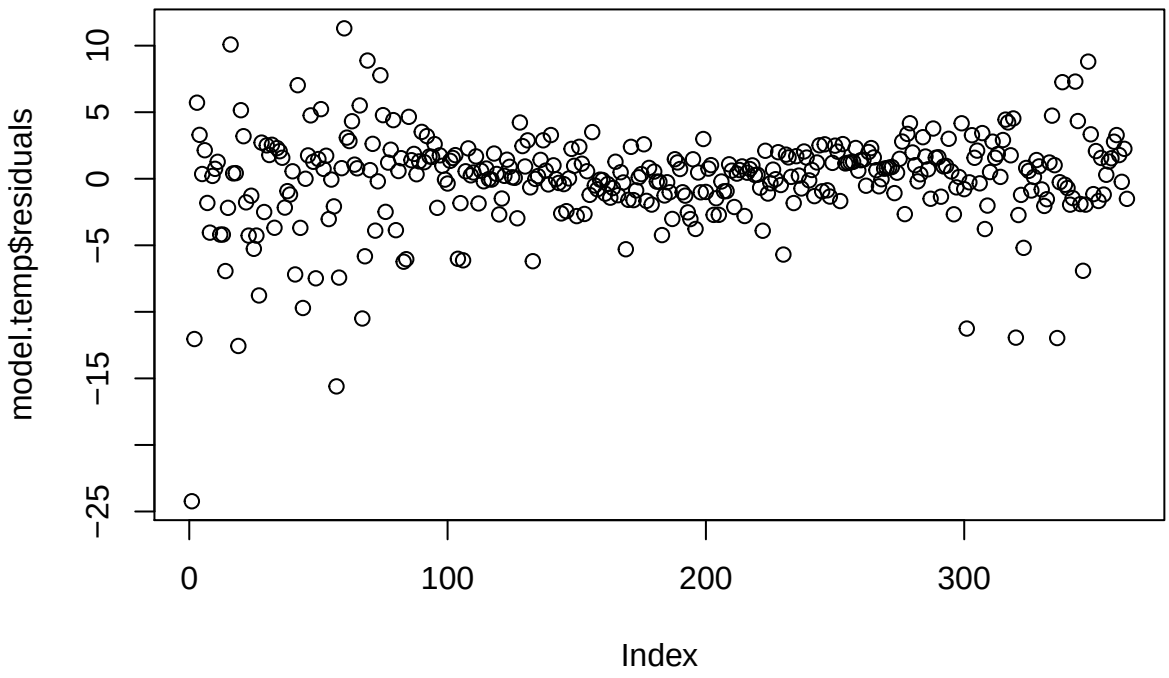
```
acf(model.temp$residuals, lag.max=30)
```



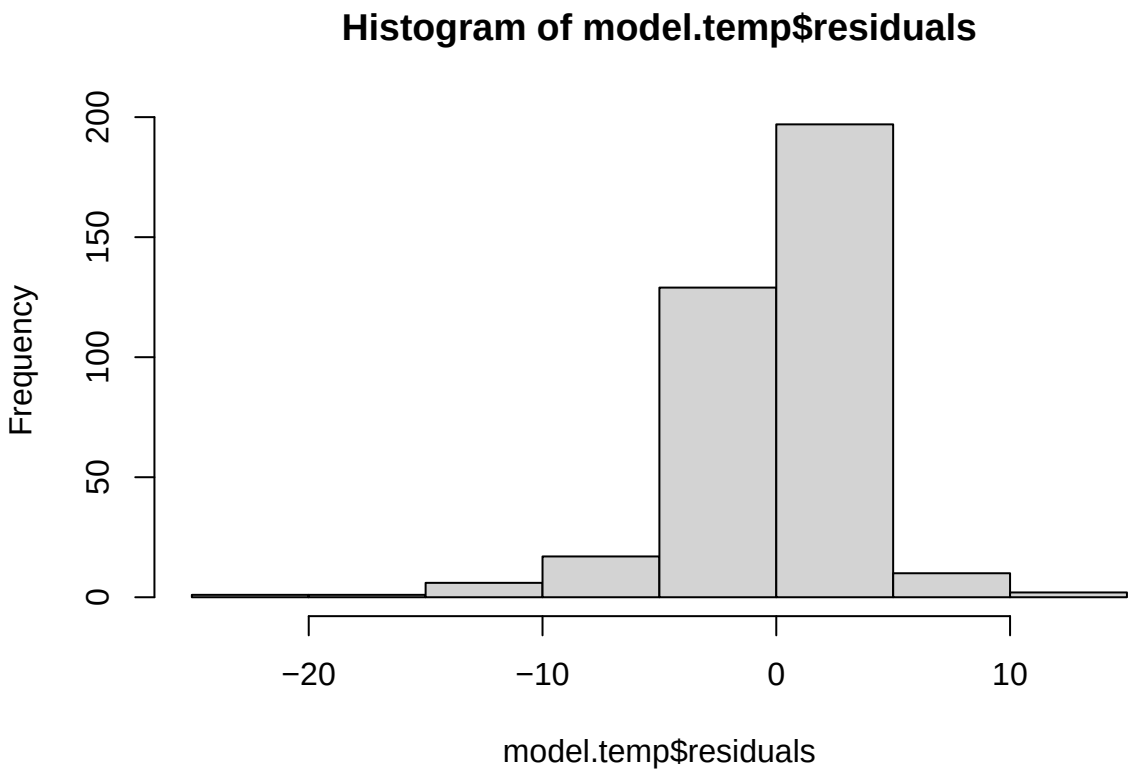

```
acf(model.temp$residuals, lag.max=365)
```




```
plot(model.temp$residuals)
```

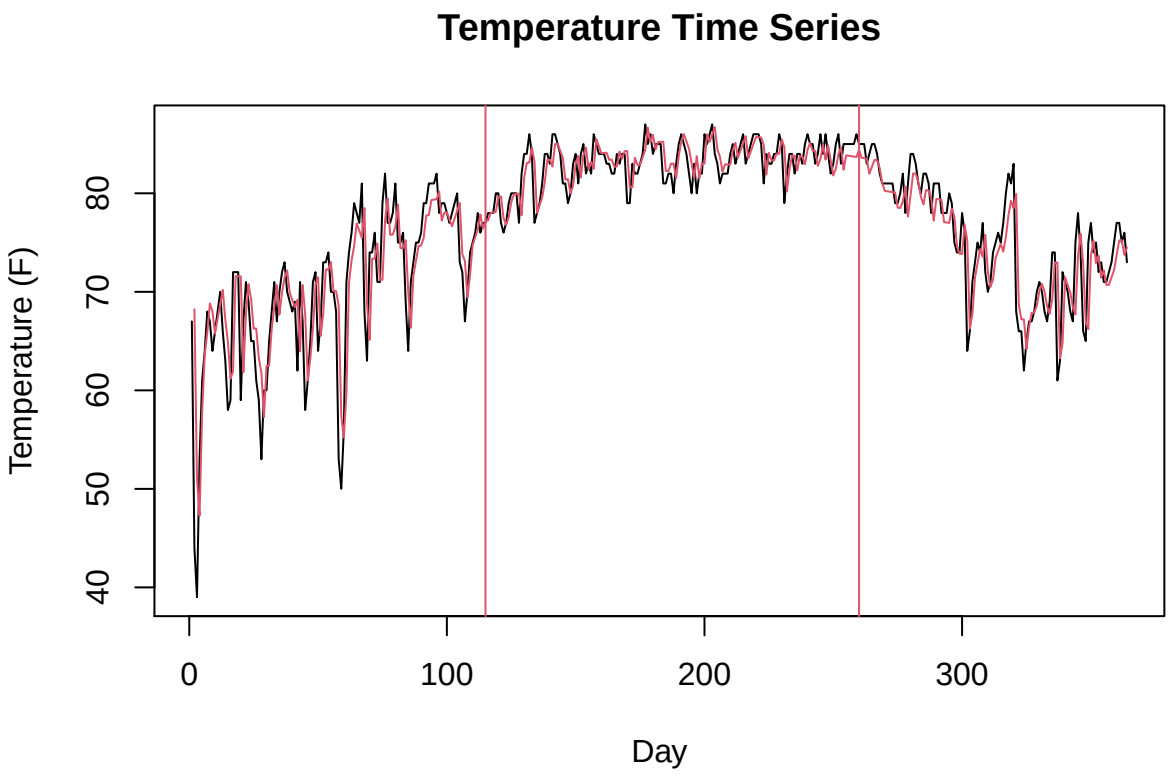



```
hist(model.temp$residuals)
```

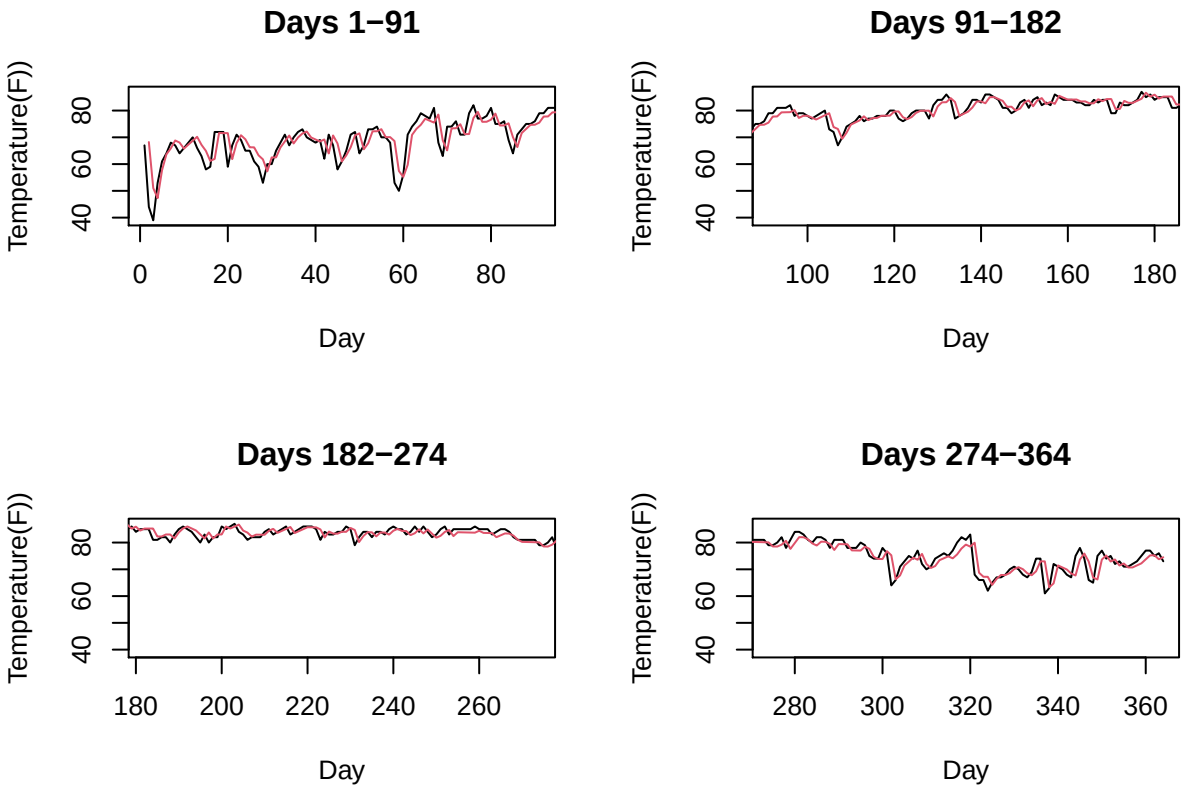


The residuals are close to normal, but there is not a constant variance, and the histogram being skewed to the left shows some overprediction of certain temperature values. I believe that this can partially be explained by acf plot of the residuals, as well as the time series of temperature itself. The acf plot for the entire year of residuals shows more tightly bound residual values that are smaller in absolute size than the rest of the year for an interval that is approximately days 100 to 230. On the time series, the amplitude of the waves for the time series are smaller for days 115 to 260. This interval is shown below with the fitted autoregressive model.

```
plot(elec$temp, type='l', xlab='Day', ylab='Temperature (F)',
     main='Temperature Time Series')

abline(v=115, col=2)
abline(v=260, col=2)

#adding line of fitted values
lines((time.l1), model.temp$fitted.values, col=2)
```




```
par(mfrow=c(2,2))
plot(elec$temp, type='l', xlab='Day', ylab = 'Temperature(F))',
     main='Days 1-91', xlim=c(1,91))
lines((time.l1), model.temp$fitted.values, col=2)

plot(elec$temp, type='l', xlab='Day', ylab = 'Temperature(F))',
     main='Days 91-182', xlim=c(91,182))
lines((time.l1), model.temp$fitted.values, col=2)

plot(elec$temp, type='l', xlab='Day', ylab = 'Temperature(F))',
     main='Days 182-274', xlim=c(182,274))
lines((time.l1), model.temp$fitted.values, col=2)

plot(elec$temp, type='l', xlab='Day', ylab = 'Temperature(F))',
     main='Days 274-364', xlim=c(274,364))
lines((time.l1), model.temp$fitted.values, col=2)
```



I believe that this is a more stable, consistently hot period of time for the climate of the region. It starts about mid-way through the month of March, and continues until about mid-September. I think a binary dummy variable as an indicator for this time period would be appropriate, and could possibly aid prediction. I will call it stable.

```
stable <- rep(FALSE, nrow(elec))
stable[115:260] <- TRUE
stable <- stable[time.l1]
stable <- as.factor(stable)
train.temp$stable <- stable
```

Since my model for temperature is already excellent looking at the plot of it overlayed with the time series, if I see satisfactory results with this new model I think it will be appropriate to go straight to forward stepwise selecion with interactions.

```
model.temp2 <- lm(cur.temp~., data=train.temp)
summary(model.temp2)
```

```
##
## Call:
## lm(formula = cur.temp ~ ., data = train.temp)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -24.2380  -1.1959   0.4493   1.6256  11.2782
```

```
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 20.17231    2.84545   7.089 7.23e-12 ***
## prev.temp    0.74387    0.03538  21.027  < 2e-16 ***
## cos310      -1.70872    0.55475  -3.080  0.00223 **
## sin310      -1.63716    0.49975  -3.276  0.00116 **
## stableTRUE  -0.45505    0.89629  -0.508  0.61198
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.513 on 358 degrees of freedom
## Multiple R-squared:  0.8089, Adjusted R-squared:  0.8068
## F-statistic: 378.9 on 4 and 358 DF,  p-value: < 2.2e-16
```

Before even proceeding, this dummy variable is clearly not significant. I am happy to use my first autoregressive model for temperature to predict future temperature values, as it had a good fit.

Forecasting

Now I will begin forecasting. Before any forecasting scheme is conducted for the variable of interest, *MW*, I need to forecast values of *temp*.

```
# time for prediction: last day of year through first week
time.pred <- 365:372

# temperaure prediction values and the 0.025 and 0.975 quantile values
# of the prediciton interval
temp.pred.frame <- data.frame( day=time.pred, val=rep(0,8),
                               lwr025 = rep(0,8), upr975 = rep(0,8))

# 8 predictions are needed, where each prediction requires
# the previous temperature value
# the first prediction can be added to my frame of values, and the rest will follow
# in a for loop
temp.pred <- predict(object = model.temp, newdata = data.frame(prev.temp=elec$temp[364],
                                                               cos310=cos(2*pi*365/300),
                                                               sin310=sin(2*pi*365/300)),
                     interval='prediction')
temp.pred.frame[1,c(2,3,4)] <- temp.pred

#in each step, create a prediction using the previously
#predicted temperature value, and add it to my frame of predictions
for(i in 2:8){
  temp.pred.frame[i,c(2,3,4)] <- predict(object = model.temp,
                                         newdata = data.frame(
                                           prev.temp=temp.pred.frame[i-1,2],
                                          cos310=cos(2*pi*time.pred[i]/300),
                                          sin310=sin(2*pi*time.pred[i]/300)),
                                         interval='prediction')
}
```

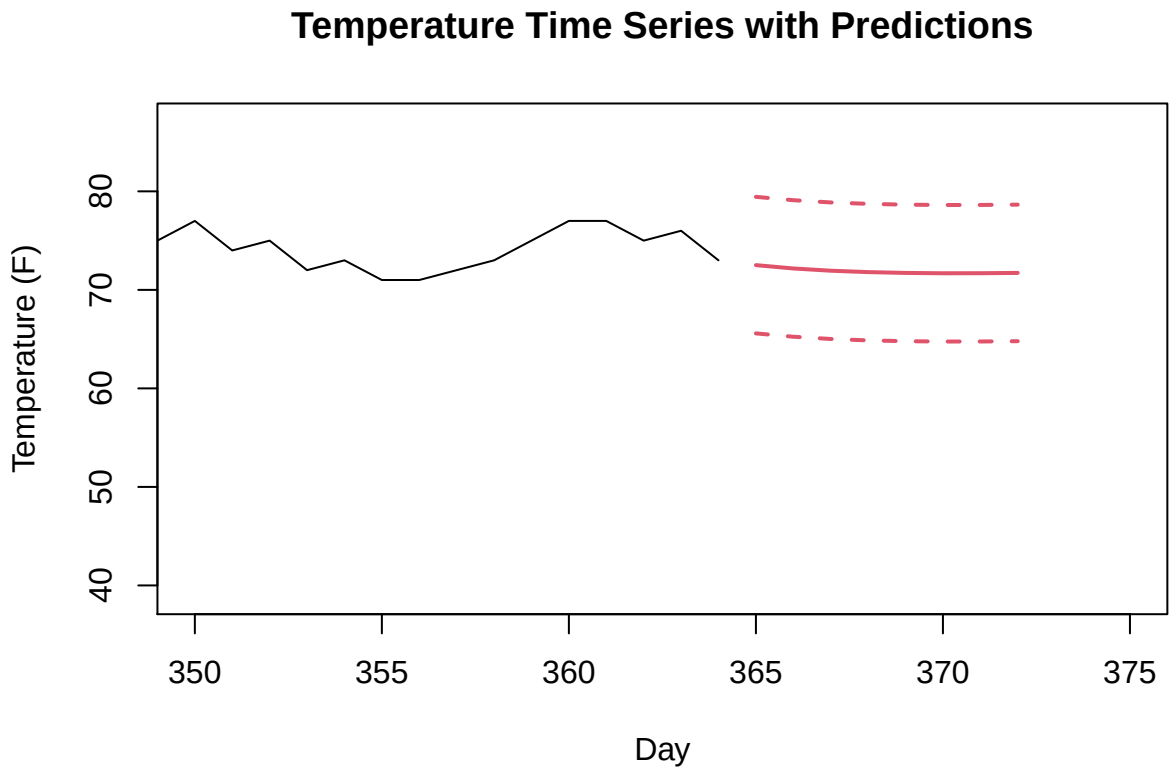
The predictions, with their upper and lower bounds for a 95% prediction interval, are displayed below.Each row represents a day, starting at 365.

```
temp.pred.frame
```

```
##   day      val   lwr025   upr975
## 1 365 72.51268 65.58237 79.44299
## 2 366 72.17346 65.24381 79.10310
## 3 367 71.94548 65.01615 78.87481
## 4 368 71.80123 64.87201 78.73045
## 5 369 71.72017 64.79094 78.64939
## 6 370 71.68697 64.75768 78.61627
## 7 371 71.69018 64.76078 78.61959
## 8 372 71.72125 64.79171 78.65078
```

And the extension of the original time series for these predictions.

```
plot(elec$temp, type='l', xlab='Day', ylab='Temperature (F)',
     main='Temperature Time Series with Predictions', xlim=c(350,375))
lines(temp.pred.frame$day, temp.pred.frame$val, col=2, lwd=2)
lines(temp.pred.frame$day, temp.pred.frame$lwr025, col=2, lty=2, lwd=2)
lines(temp.pred.frame$day, temp.pred.frame$upr975, col=2, lty=2, lwd=2)
```



Now that temperature values have been obtained, I can construct the appropriate explanatory variables for forecasting the MW values. In order to measure the uncertainty in the forecasted values, I will do 3 sets of forecasts. These sets of forecasts will use the predicted temperatures, the upper bounds of each prediction, and the lower bounds. For each set of predictions, there will also be 95% prediction intervals giving overall uncertainty. This strategy will be more apparent once the predictions are visualized.

I will start by building three data frames for each sets of forecasts. The temperature ranges are uniform for each forecast set. The mean prediction values and lower bounds are in the 'med' temperature range, where my upper values are in the 'high' temperature range.

```
# time interval for predictions
time.pred <- 365:372
# weekend values for my forecasted days, which starts on a sunday
weeknd <- c(TRUE, FALSE, FALSE, FALSE, FALSE, FALSE, TRUE, TRUE)
# using predicted temperature and other explanatory variables
pred.MW.mid.frame <- data.frame(temp = temp.pred.frame$val,
                                temper.range=as.factor(rep('med',8)),
                                weekend = as.factor(weeknd))
# using upperbound temperature and other explanatory variables
pred.MW.upper.frame <- data.frame(temp = temp.pred.frame$upr975,
                                  temper.range=as.factor(rep('high',8)),
                                  weekend = as.factor(weeknd))
# using lower bound temperature and other explanatory variables
pred.MW.lower.frame <- data.frame(temp = temp.pred.frame$lwr025,
                                  temper.range=as.factor(rep('med',8)),
                                  weekend = as.factor(weeknd))

# these data frames are for my results
results.forecast.mid <- data.frame(day=time.pred, val=rep(0,8),
                                   lwr025 = rep(0,8), upr975=rep(0,8))
results.forecast.lwr <- data.frame(day=time.pred, val=rep(0,8),
                                   lwr025 = rep(0,8), upr975=rep(0,8))
results.forecast.upr <- data.frame(day=time.pred, val=rep(0,8),
                                   lwr025 = rep(0,8), upr975=rep(0,8))

# now, using the predict function, I can get my results
mid.predictions <- predict(object=model.MLR2, newdata=pred.MW.mid.frame
                           , interval='prediction')
upr.predictions <- predict(object=model.MLR2, newdata=pred.MW.upper.frame,
                           interval='prediction')
lwr.predictions <- predict(object=model.MLR2, newdata=pred.MW.lower.frame,
                           interval='prediction')

#putting results in my frames
results.forecast.mid[,2:4] <- mid.predictions[,1:3]
results.forecast.lwr[,2:4] <- lwr.predictions[,1:3]
results.forecast.upr[,2:4] <- upr.predictions[,1:3]

# an example of what these dataframes look like
results.forecast.upr
```

```
##   day      val   lwr025   upr975
## 1 365 3169.488 2807.562 3531.414
## 2 366 3332.702 2971.764 3693.640
## 3 367 3305.120 2944.106 3666.134
## 4 368 3287.678 2926.613 3648.744
## 5 369 3277.886 2916.790 3638.982
## 6 370 3273.883 2912.775 3634.992
## 7 371 3070.005 2707.803 3432.207
## 8 372 3073.774 2711.584 3435.964
```

Displayed above is the set of MW forecasts for the upperbound of the prediction interval of temperature. The the sets for the fitted temperature predictions, then the lower bound of the temperature prediction intervals and the associated MW forecasts are displayed below.

```
results.forecast.mid
```

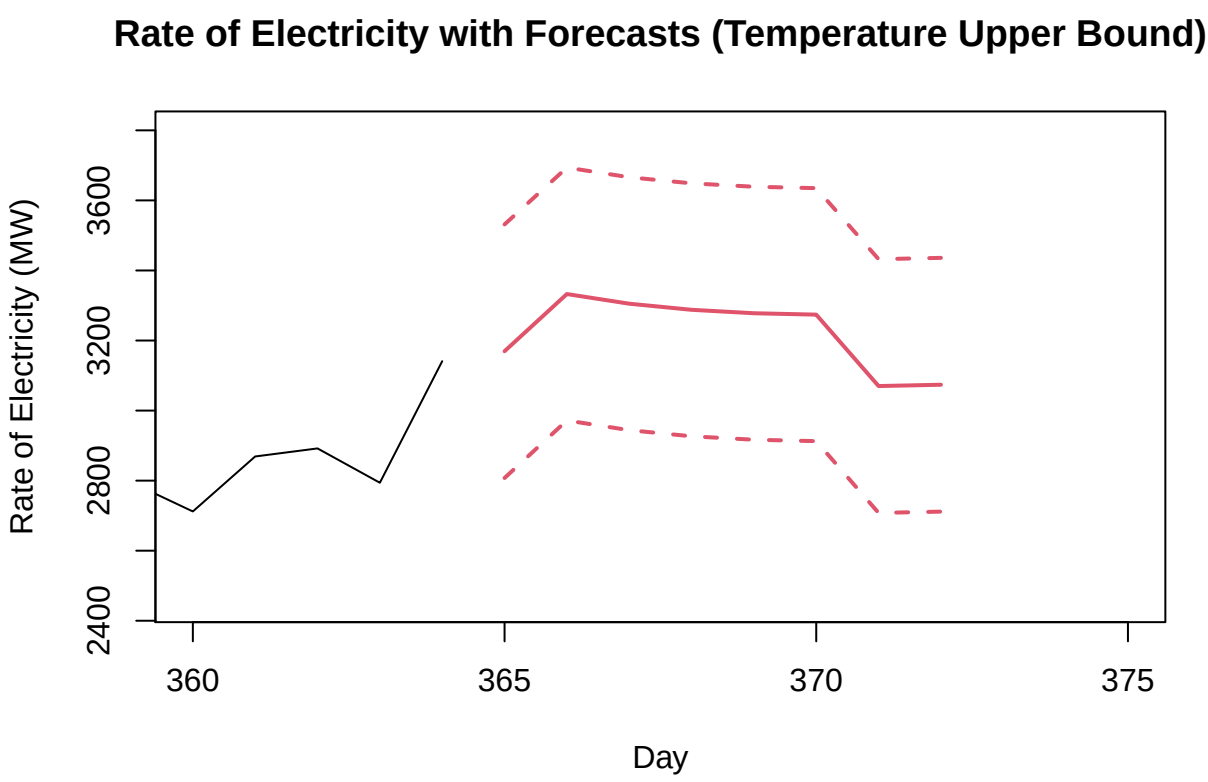
```
##   day      val   lwr025   upr975
## 1 365 2924.069 2560.059 3288.078
## 2 366 3128.482 2765.734 3491.231
## 3 367 3128.572 2765.989 3491.156
## 4 368 3128.629 2766.145 3491.114
## 5 369 3128.661 2766.230 3491.093
## 6 370 3128.675 2766.264 3491.085
## 7 371 2924.394 2560.995 3287.793
## 8 372 2924.382 2560.962 3287.801
```

```
results.forecast.lwr
```

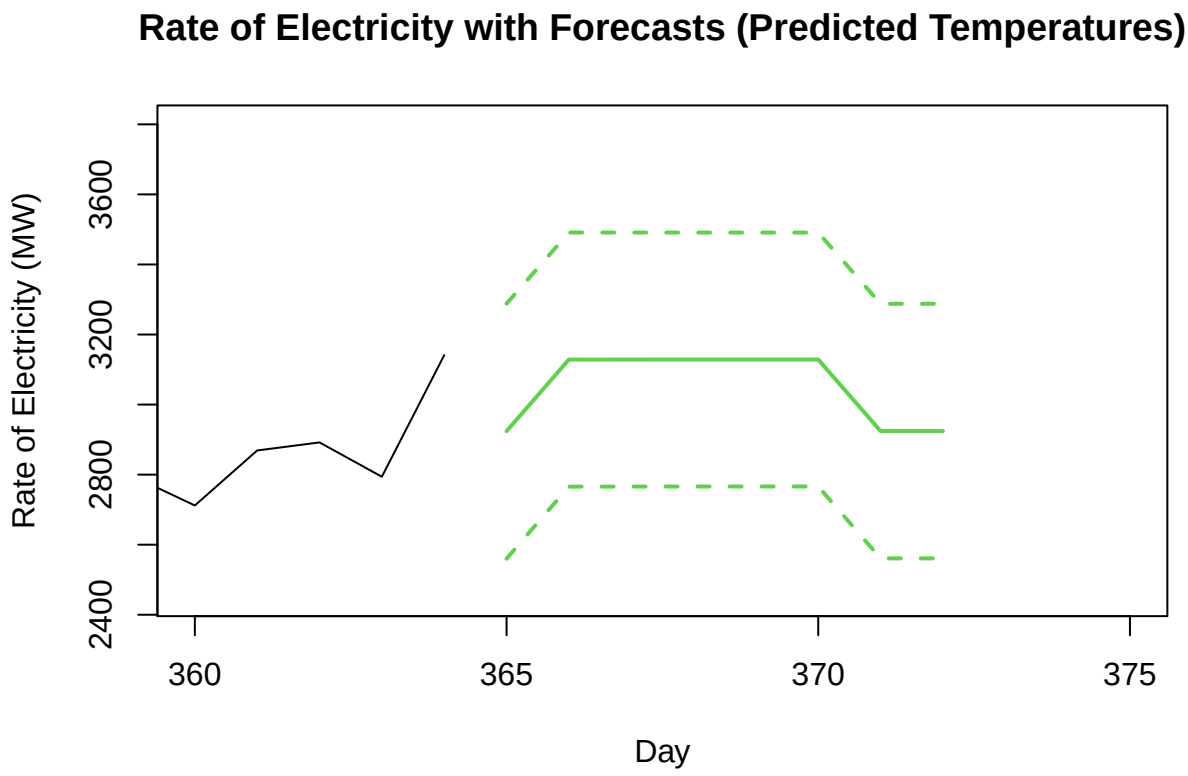
```
##   day      val   lwr025   upr975
## 1 365 2926.808 2563.141 3290.475
## 2 366 3131.221 2768.208 3494.235
## 3 367 3131.311 2768.104 3494.519
## 4 368 3131.368 2768.032 3494.704
## 5 369 3131.400 2767.990 3494.811
## 6 370 3131.413 2767.972 3494.855
## 7 371 2927.133 2562.784 3291.481
## 8 372 2927.121 2562.801 3291.441
```

Let's visualize the results, starting with 3 plots of a continued time series of MW values for each of the sets of forecasts.

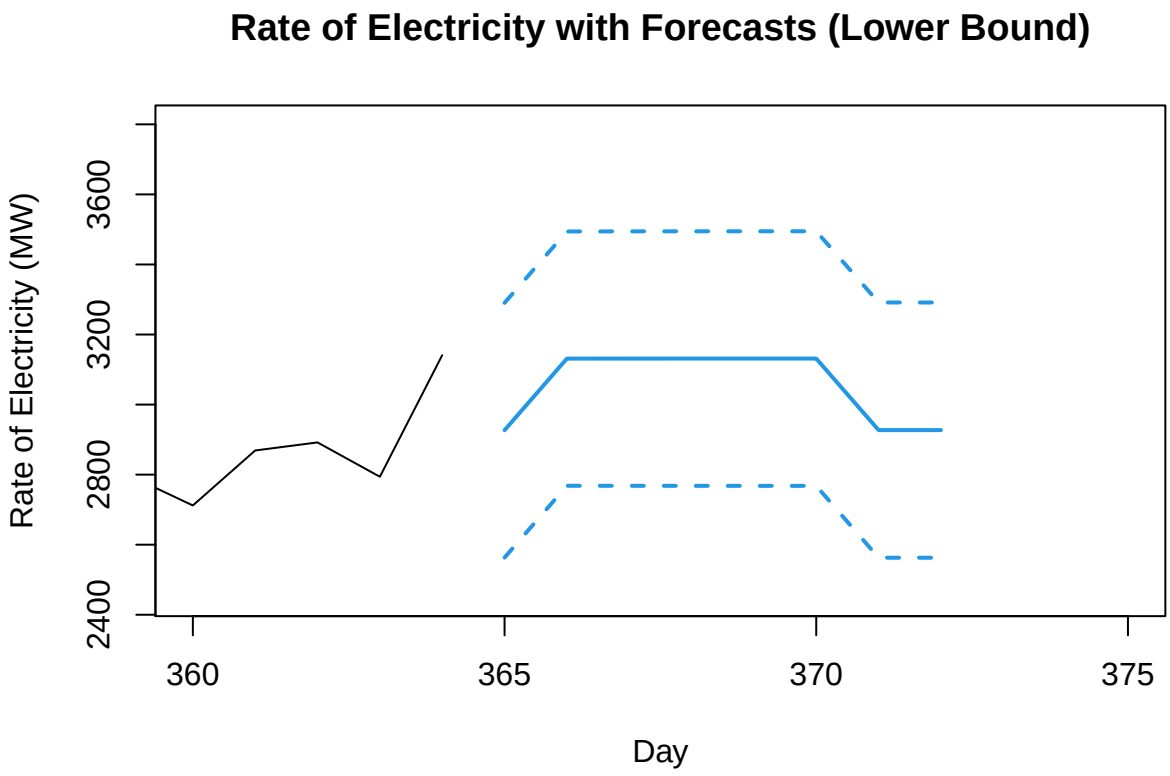
```
# upper temps
plot(elec$MW, type='l', xlab='Day', ylab='Rate of Electricity (MW)',
     main='Rate of Electricity with Forecasts (Temperature Upper Bound)',
     xlim=c(360,375), ylim=c(2450, 3800))
lines(temp.pred.frame$day, results.forecast.upr$val, col=2, lwd=2)
lines(temp.pred.frame$day, results.forecast.upr$lwr025, col=2, lty=2, lwd=2)
lines(temp.pred.frame$day, results.forecast.upr$upr975, col=2, lty=2, lwd=2)
```




```
#fit temps
plot(elec$MW, type='l', xlab='Day', ylab='Rate of Electricity (MW)',
     main='Rate of Electricity with Forecasts (Predicted Temperatures)',
     xlim=c(360,375), ylim=c(2450, 3800))
lines(temp.pred.frame$day, results.forecast.mid$val, col=3, lwd=2)
lines(temp.pred.frame$day, results.forecast.mid$lwr025, col=3, lty=2, lwd=2)
lines(temp.pred.frame$day, results.forecast.mid$upr975, col=3, lty=2, lwd=2)
```

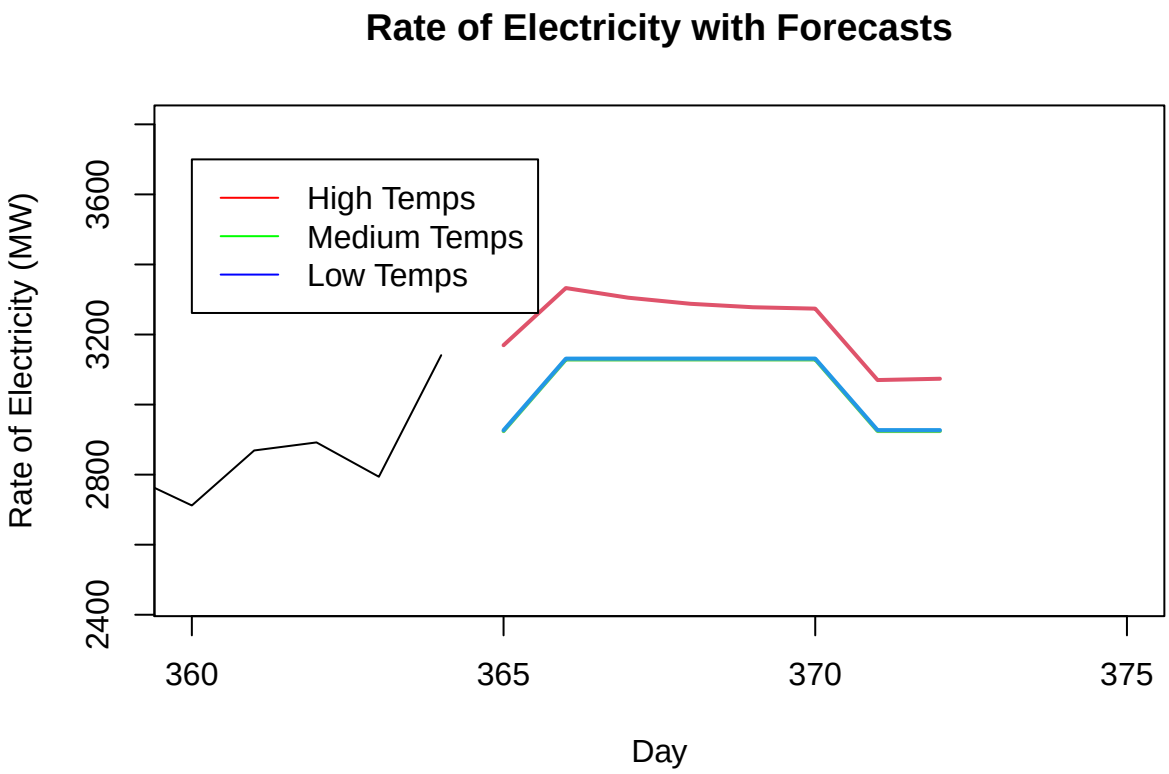



```
#lower temps
plot(elec$MW, type='l', xlab='Day', ylab='Rate of Electricity (MW)',
     main='Rate of Electricity with Forecasts (Lower Bound)',
     xlim=c(360,375), ylim=c(2450, 3800))
lines(temp.pred.frame$day, results.forecast.lwr$val, col=4, lwd=2)
lines(temp.pred.frame$day, results.forecast.lwr$lwr025, col=4, lty=2, lwd=2)
lines(temp.pred.frame$day, results.forecast.lwr$upr975, col=4, lty=2, lwd=2)
```



An overlay of the fitted MW values from each of the three forecast sets.

```
plot(elec$MW, type='l', xlab='Day', ylab='Rate of Electricity (MW)',
     main='Rate of Electricity with Forecasts', xlim=c(360,375), ylim=c(2450, 3800))
lines(temp.pred.frame$day, results.forecast.upr$val, col=2, lwd=2)
lines(temp.pred.frame$day, results.forecast.mid$val, col=3, lwd=2)
lines(temp.pred.frame$day, results.forecast.lwr$val, col=4, lwd=2)
legend(360, 3700, legend = c('High Temps', 'Medium Temps', 'Low Temps'),
       col = c('red', 'green', 'blue'), lty = 1)
```

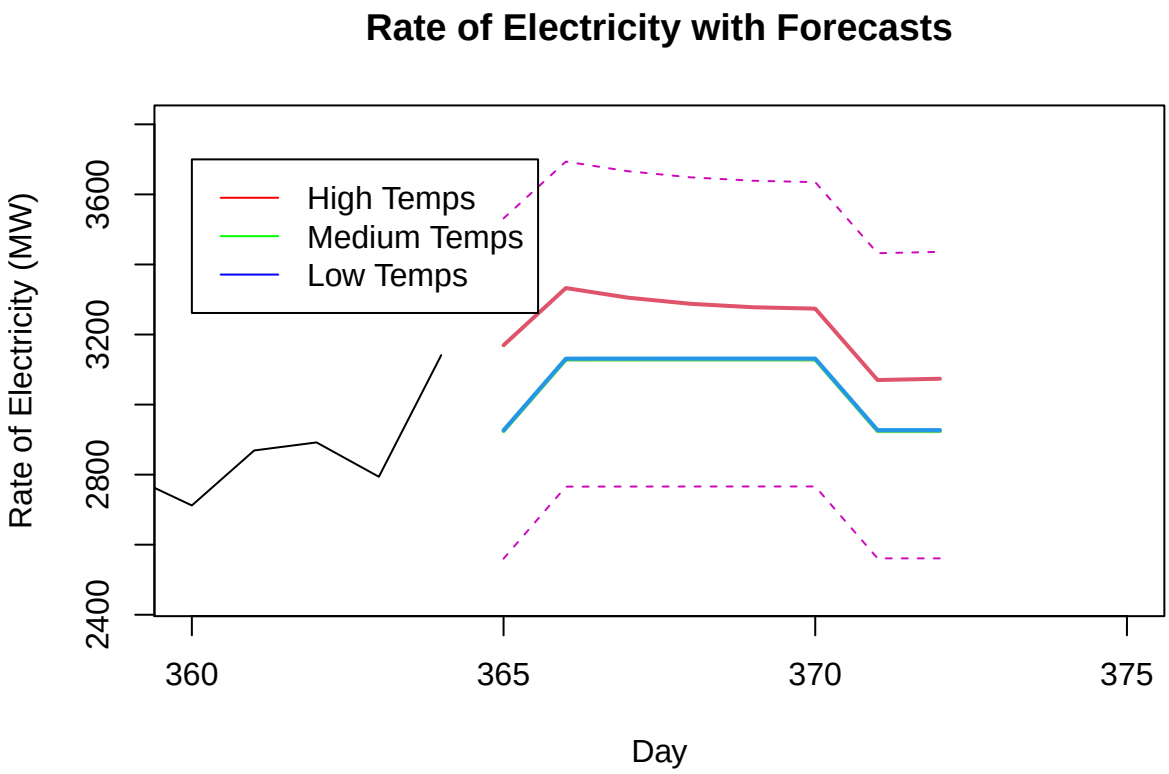


It will be valuable to provide the highest prediction boundary of the high temp forecasts (which I can visually determine is larger values than the other 2 forecast sets), and the lowest of the fitted temp for low temp forecasts. Let's figure out which of these boundaries is lower, and plot it.

```
results.forecast.mid$lwr025 - results.forecast.lwr$lwr025
```

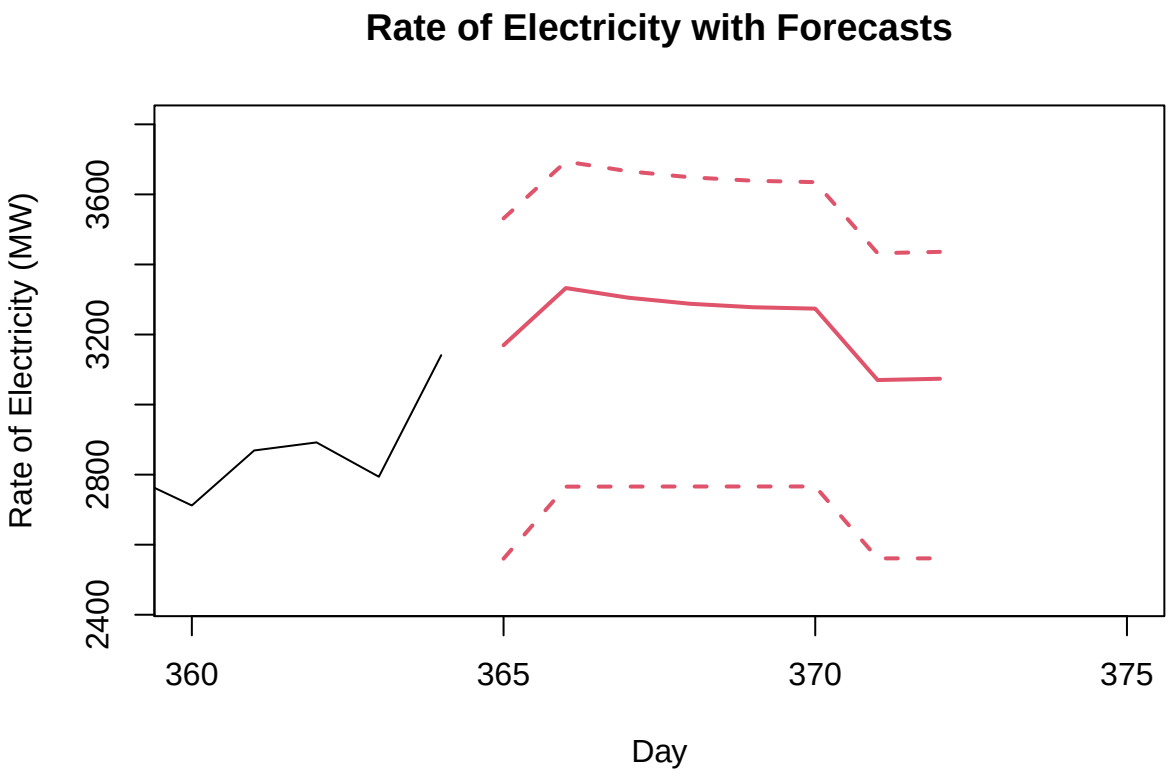
```
## [1] -3.082073 -2.474281 -2.115059 -1.887768 -1.760030 -1.707681 -1.789318
## [8] -1.838076
```

```
# the output shows the lower bounds for mid temperatures are lower than the low temperatures.
# they will serve as the lower bounds for the overall forecasts.

plot(elec$MW, type='l', xlab='Day', ylab='Rate of Electricity (MW)',
     main='Rate of Electricity with Forecasts', xlim=c(360,375), ylim=c(2450, 3800))
lines(temp.pred.frame$day, results.forecast.upr$val, col=2, lwd=2)
lines(temp.pred.frame$day, results.forecast.mid$val, col=3, lwd=2)
lines(temp.pred.frame$day, results.forecast.lwr$val, col=4, lwd=2)
lines(temp.pred.frame$day, results.forecast.mid$lwr025, col=6, lty=2)
lines(temp.pred.frame$day, results.forecast.upr$upr975, col=6, lty=2)
legend(360, 3700, legend = c('High Temps', 'Medium Temps', 'Low Temps'),
       col = c('red', 'green', 'blue'), lty = 1)
```



All that remains is to decide on my final MW values that will serve as my forecasted set, as I have 3 sets of forecasts. I think that using the forecast set from the fitted temperature predictions will be best. They are the expected temperatures from my model. To calculate uncertainty, I will use the bound displayed above. Here is a final visualization and table of my findings.

```
results.final <- data.frame(Day = 365:372,
                            Forecasted_Electric_Rate = results.forecast.mid$val,
                            Prediction_Interval_Upper_Bound = results.forecast.upr$upr975,
                            Prediction_Interval_Lower_Bound = results.forecast.mid$lwr025)

plot(elec$MW, type='l', xlab='Day', ylab='Rate of Electricity (MW)',
     main='Rate of Electricity with Forecasts', xlim=c(360,375), ylim=c(2450, 3800))
lines(temp.pred.frame$day, results.forecast.upr$val, col=2, lwd=2)
lines(temp.pred.frame$day, results.forecast.mid$lwr025, col=2, lty=2, lwd=2)
lines(temp.pred.frame$day, results.forecast.upr$upr975, col=2, lty=2, lwd=2)
```




```
results.final
```

```
##   Day Forecasted_Electric_Rate Prediction_Interval_Upper_Bound
## 1 365                 2924.069                         3531.414
## 2 366                 3128.482                         3693.640
## 3 367                 3128.572                         3666.134
## 4 368                 3128.629                         3648.744
## 5 369                 3128.661                         3638.982
## 6 370                 3128.675                         3634.992
## 7 371                 2924.394                         3432.207
## 8 372                 2924.382                         3435.964
##   Prediction_Interval_Lower_Bound
## 1                        2560.059
## 2                        2765.734
## 3                        2765.989
## 4                        2766.145
## 5                        2766.230
## 6                        2766.264
## 7                        2560.995
## 8                        2560.962
```

Conclusion

After analysis I have determined that there are indeed predictors that allow a model of electricity demand. The temperature of a day has a strong determination of electricity demand. Specifically, this temperature,

and the range of temperature that it is apart of, together affect electicity demand. Three different regression lines can be fit for three temperature ranges. Low temperatures are those less than 60 degrees Fahrenheit, medium between 60 and 74 degrees, and high greater than 74 degrees. In the low temperature range, as temperature decreases, electricity demand increases. For the medium temperature range, there is a slight decrease in electricity demand as temperature increases but the electricity is mostly consistent. For the high temperature range, as temperature increases, the electricity demand increases. Additionally, there is less electricity demand on weekends, demonstrated by the negative regression coefficient for this feature that I created.

I believe that the electricity demand can be accurately forecasted. I was able to produce a well fitting temperature time series model, and used those values to forecast a very tight fitting multiple regression model for electricity demand. I was able to get 3 seperate forecasts for electricity demand dependent on the mean or bounds of the forecasted electricity for each day, and even when taking the maximum range of electricity demand in the forecast, the range of electricity demand was roughly 1000 MW at most for the day. Given the limited number of variables available, I consider this an accurate forecast.

Titanic Dataset

Summary

In 1912 the now infamous sinking of the titanic occured. More than 2/3 of the passengers onboard ultimately perished from the event. Using data provided by Kaggle at www.kaggle.com/c/titanic/data, analysis can be done to help understand which passengers were more likely to survive the incident than others. The dataset from Kaggle has information relating to most of the passengers onboard the ship with fields like Name, Ticket Price (Fare), port of embarkment, Sex, number of parents or children onboard with the passenger, among other fields. Is this information enough to accurately classify the passengers as having survived the incident or not? And if so, which of these fields are stronger predictors than others?

In my analysis, I was able to get above-average performance on a classifier. I elected to use a logistic regression model, which uses a regression model for binary classification. This regression model outputs the probability of 'success' (in this dataset survival) based on the variables (passenger information) that the model is constructed with. In order to get the level of performance that my model achieved, I had to engineer some features from the provided passenger information. These features were mainly taking count data like number of parents or children onboard, and number of siblings and spouses onboard, and engineering features for a subset of these counts. For example, instead of using the field number of parents or children onboard as a continous variable (or even a categorical variable), my EDA found that creating a dummy variable to indicate whether a passenger had at least 1 parent or child onboard would be more valuable.

After a few iterations of building a logistic regression model with different fields, I decided on a simple model with more than 10 variables, but no interactions amongst the variables. This model was trained on 85% of the provided non-testing data, and validated on the remaining 15%. The model had a Bayesion Information Criterion (BIC) score of 729.8, a Brier Score (mean squared difference of predicted probability of survival and true outcome (1 or 0)) of 0.1278, an accuracy (proportion of correct predictions to total predictions) of 82.84%, an F1-score (harmonic mean of True Positive Rate and proportion of true positives to predicted positives) of 0.7629, and a specificity (True Negative Rate) of 87.06%. When tested on the test dataset of passengers, 154 of them were classified as survivors. The training and validation set had a total of 342 survivors, and the total number of passengers that survived is about 500. This means that an expectation of about 158 survivors is the number of survivors in the test dataset. My classifier missed this mark by only 4. After testing, the classifications for passengers by group were observed. **The biggest finding, and the most glaring highlight in this whole analysis, is that of the 100 passengers in my test dataset who were men with the title Mr. that purchased 3rd class tickets, without spouses, siblings, parents or children on board, every single one was classified as not surviving.**

Data Analysis

EDA Phase

Let's start by loading the datasets for this problem, and viewing a summary of the features as well as the head of the dataset.

```
data_titanic <- read.csv('titanic_imputted.csv')
titanic_test <- read.csv('titanic_imputed_test.csv')
summary(data_titanic)
```

```
##   PassengerId       Survived          Pclass          Name
##  Min.   :  1.0   Min.   :0.0000   Min.   :1.000   Length:891
##  1st Qu.:223.5   1st Qu.:0.0000   1st Qu.:2.000   Class :character
##  Median :446.0   Median :0.0000   Median :3.000   Mode  :character
##  Mean   :446.0   Mean   :0.3838   Mean   :2.309
##  3rd Qu.:668.5   3rd Qu.:1.0000   3rd Qu.:3.000
##  Max.   :891.0   Max.   :1.0000   Max.   :3.000
##      Sex                 Age            SibSp           Parch
##  Length:891         Min.   : 0.42   Min.   :0.000   Min.   :0.0000
##  Class :character   1st Qu.:21.00   1st Qu.:0.000   1st Qu.:0.0000
##  Mode  :character   Median :29.00   Median :0.000   Median :0.0000
##                     Mean   :29.91   Mean   :0.523   Mean   :0.3816
##                     3rd Qu.:38.00   3rd Qu.:1.000   3rd Qu.:0.0000
##                     Max.   :80.00   Max.   :8.000   Max.   :6.0000
##     Ticket               Fare           Cabin              Embarked
##  Length:891         Min.   :  0.00   Length:891         Length:891
##  Class :character   1st Qu.:  7.91   Class :character   Class :character
##  Mode  :character   Median : 14.45   Mode  :character   Mode  :character
##                     Mean   : 32.20
##                     3rd Qu.: 31.00
##                     Max.   :512.33
##     Title              AgeEst
##  Length:891         Mode :logical
##  Class :character   FALSE:714
##  Mode  :character   TRUE :177
##
##
##
```

```
head(data_titanic)
```

```
##   PassengerId Survived Pclass
## 1           1        0      3
## 2           2        1      1
## 3           3        1      3
## 4           4        1      1
## 5           5        0      3
## 6           6        0      3
##                                                  Name    Sex Age SibSp Parch
## 1                             Braund, Mr. Owen Harris   male  22     1     0
## 2 Cumings, Mrs. John Bradley (Florence Briggs Thayer) female  38     1     0
## 3                              Heikkinen, Miss. Laina female  26     0     0
```

```
## 4        Futrelle, Mrs. Jacques Heath (Lily May Peel) female  35     1     0
## 5                             Allen, Mr. William Henry   male  35     0     0
## 6                                     Moran, Mr. James   male  22     0     0
##             Ticket    Fare Cabin Embarked Title AgeEst
## 1        A/5 21171  7.2500              S    Mr  FALSE
## 2         PC 17599 71.2833   C85        C   Mrs  FALSE
## 3 STON/O2. 3101282  7.9250              S  Miss  FALSE
## 4           113803 53.1000  C123        S   Mrs  FALSE
## 5           373450  8.0500              S    Mr  FALSE
## 6           330877  8.4583              Q    Mr   TRUE
```

Since the model I have in mind for this problem is a binary classification model, I already see certain features that will not be of use. For example, the ticket variable will likely not be useful. It is not a continous variable, and converting it to a dummy variable will make way too many levels.

The names will also not be of value, as the titles in the names are already a feature. The titles themselves may not be that valuable since there is a feature for sex already present. Lastly, the cabin feature may not be that valuable since there are missing values. Let's explore some of these potentially problematic features.

```
# number of levels for categorical Ticket Variable
length(levels(as.factor(data_titanic$Ticket)))
```

```
## [1] 681
```

```
# number of levels for cabin variable
length(levels(as.factor(data_titanic$Cabin)))
```

```
## [1] 148
```

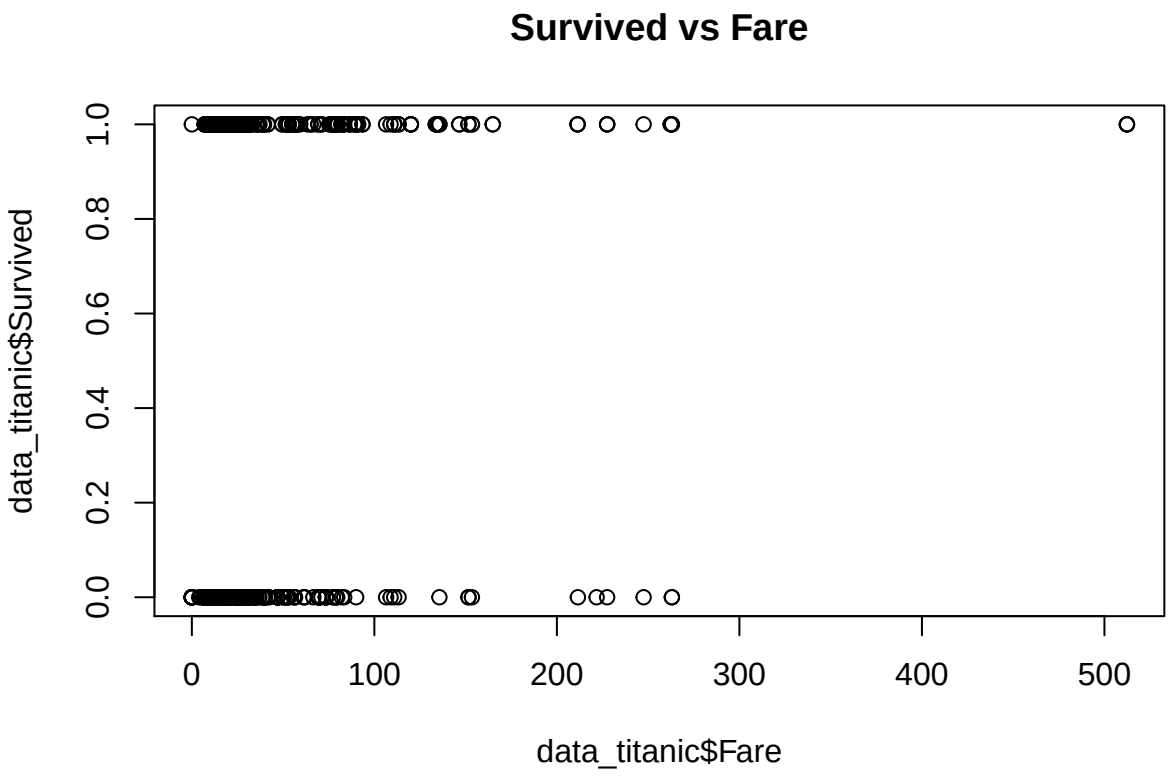
```
# levels for title
levels(as.factor(data_titanic$Title))
```

```
## [1] "Master" "Miss"   "Mr"     "Mrs"    "Other"
```

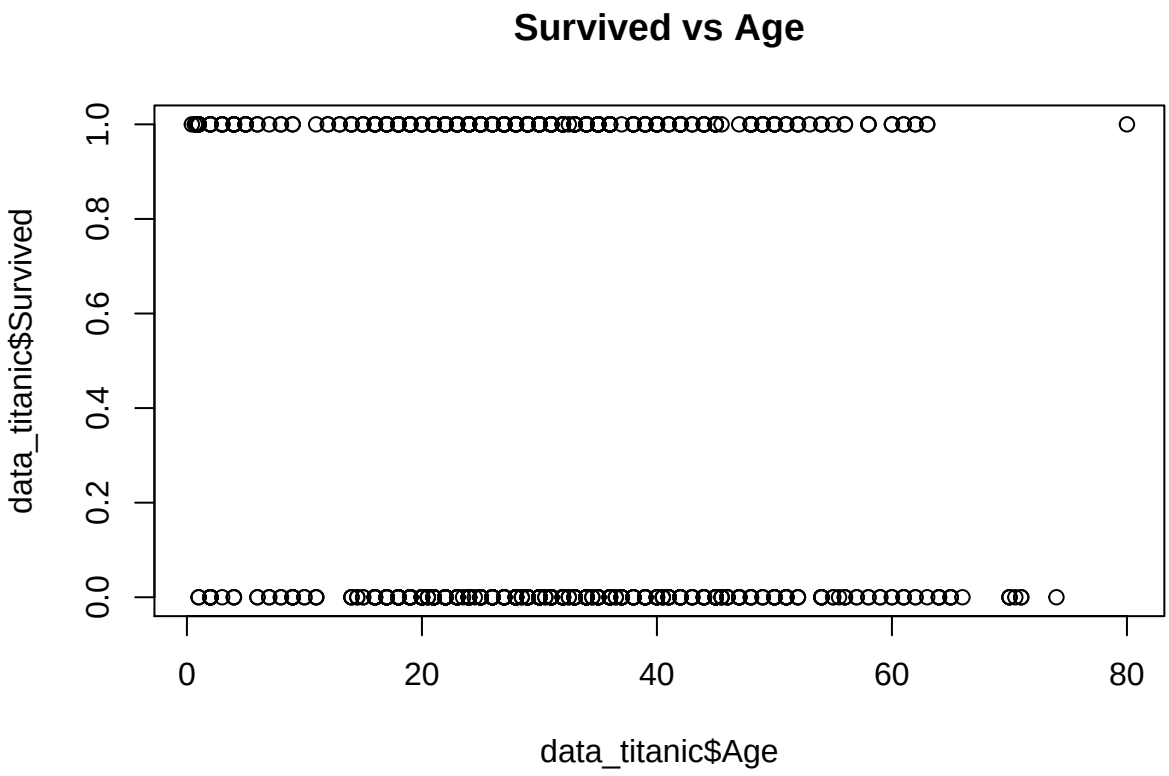
The Ticket and Cabin variables will not be useful with so many variables. However, for now, the Title variable might be of use and there few enough variables that they wouldn't results in overfitting problems for later validation and testing.

Now, I'm going to plot some of the features that will likely be useful. With the dependent variable survived being a binary indicator, I would like to see a plot of survived versus some of the continous variables provided in the dataset. Fare and Age are the 2 continous variables provided.

```
# plot of Survived vs Fare
plot(data_titanic$Fare, data_titanic$Survived, main='Survived vs Fare')
```



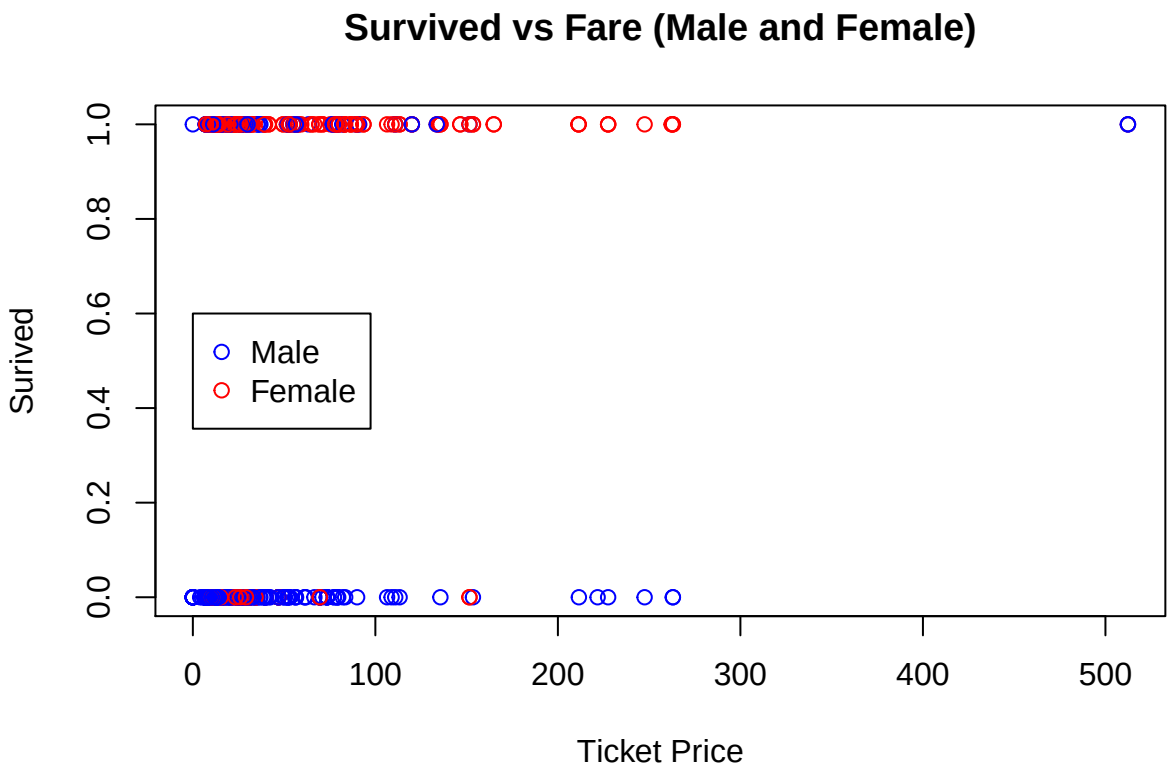

```r
# plot of Survived vs Age
plot(data_titanic$Age, data_titanic$Survived, main ='Survived vs Age')
```



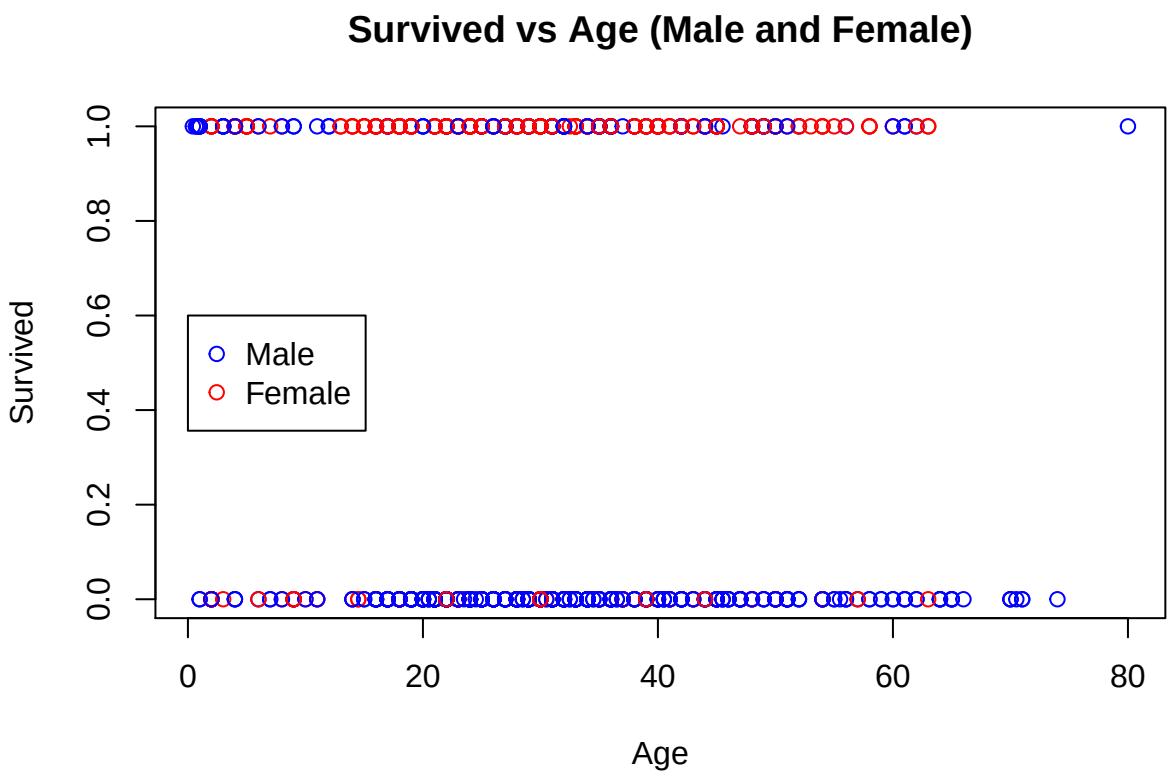
For the plot Survived vs Fare, It appears that as ticket price increased, there was at least some degree of increased probability of survival for the more expensive ticket prices. Additionally, there is one outlier for Fare that would be problematic for training a model. The Survived vs Age plot does not appear to be very useful, but there does seem to be a bit of grouping for survival for the very young passengers (babies and infants with age less than 2).

I think adding a second covariate to these plots will give a bit more insight. Let's color the plots with their sex categorization.

```
# colors where blue is for male, red female
sex_color <- ifelse(data_titanic$Sex == 'male','blue', 'red')

# survived vs fare
plot(data_titanic$Fare, data_titanic$Survived, col=sex_color,
     main = 'Survived vs Fare (Male and Female)', xlab='Ticket Price', ylab='Surived')
legend(0, 0.6, legend = c('Male', 'Female'), col = c('blue', 'red'), pch=1)
```



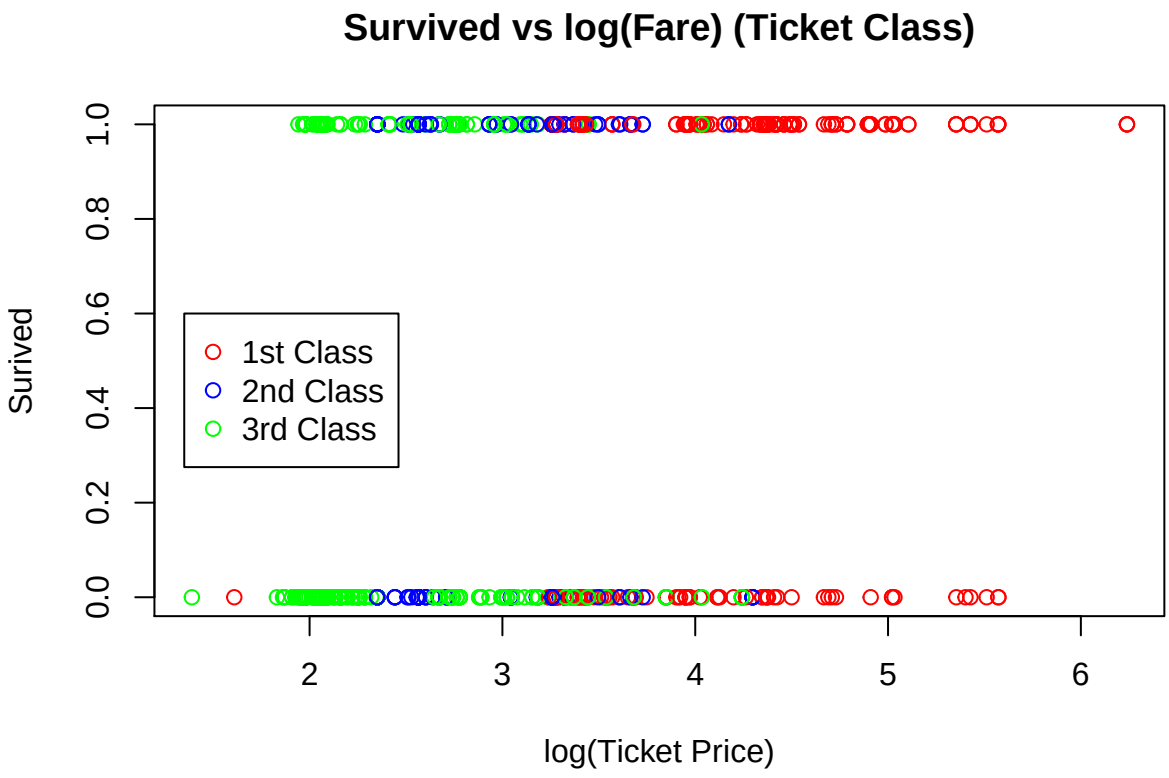

```
# survived vs age
plot(data_titanic$Age, data_titanic$Survived, col= sex_color,
     main='Survived vs Age (Male and Female)', xlab='Age', ylab = 'Survived')
legend(0, 0.6, legend = c('Male', 'Female'), col = c('blue', 'red'), pch=1)
```



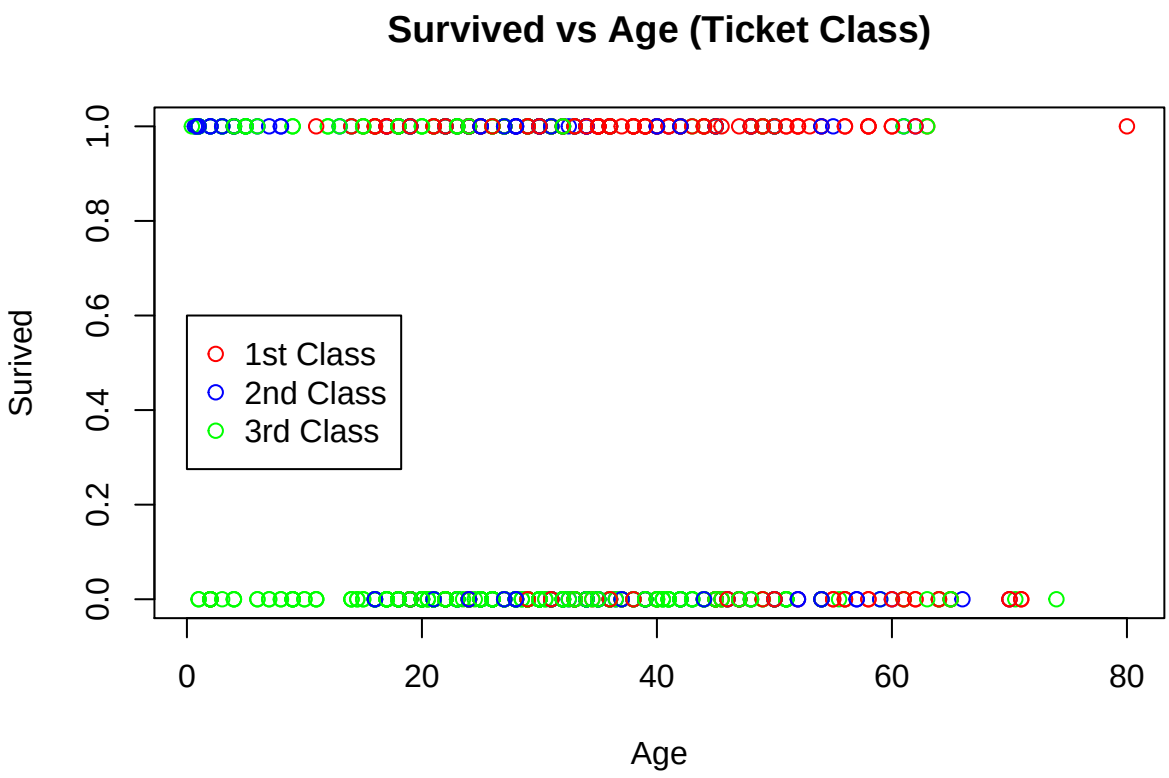
These plots are extremely revealing. The passenger's sex is extremley impactful on their probability of survival.

I am going to repeat this process, except looking at the class of the passengers' tickets provided by the Pclass variable. In this plot, I will take the log of the Fare variable to see if it 'brings in' some of the variables that might be outliers.

```
pclass_color <- ifelse(data_titanic$Pclass==1, 'red', 'blue')
pclass_color[data_titanic$Pclass==3] <- 'green'

# survived vs fare
plot(log(data_titanic$Fare), data_titanic$Survived, col=pclass_color,
     main = 'Survived vs log(Fare) (Ticket Class)', xlab='log(Ticket Price)',
     ylab='Surived')

legend(1.35, 0.6, legend = c('1st Class', '2nd Class', '3rd Class'),
       col = c('red', 'blue', 'green'), pch=1)
```



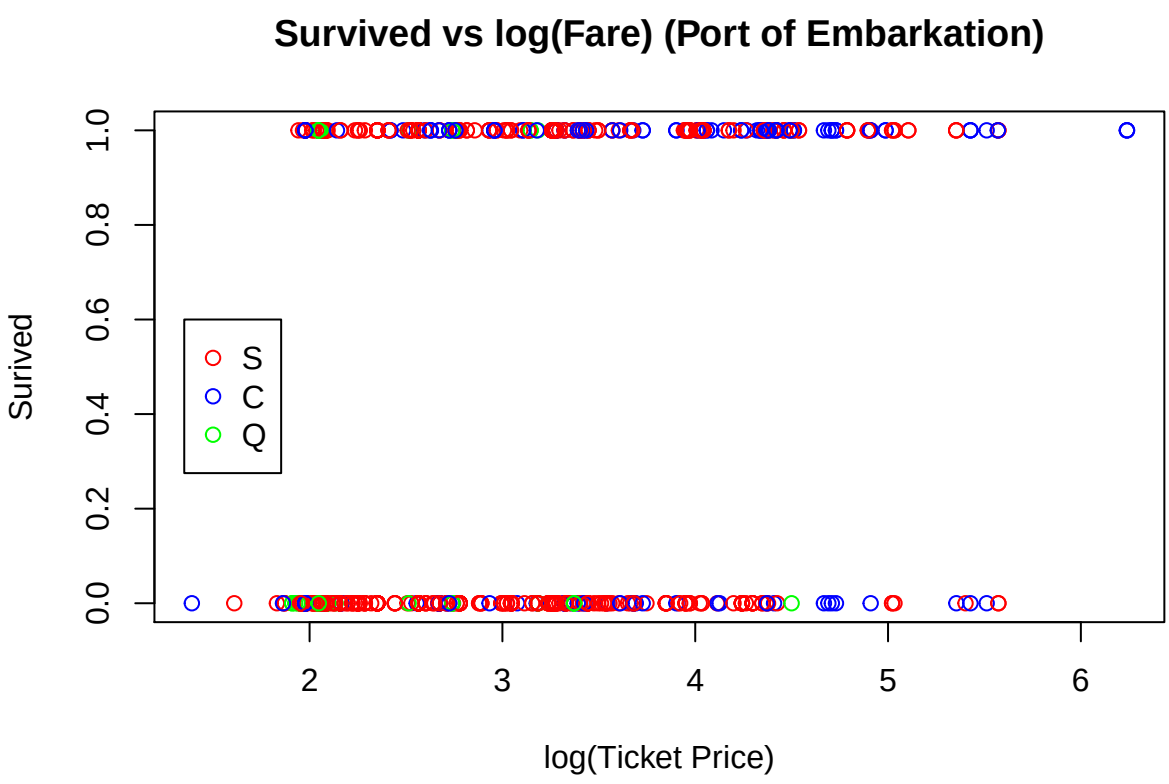

```
# survived vs age
plot(data_titanic$Age, data_titanic$Survived, col= pclass_color,
     main='Survived vs Age (Ticket Class)', xlab='Age',
     ylab='Surived')

legend(0, 0.6, legend = c('1st Class', '2nd Class', '3rd Class'),
       col = c('red', 'blue', 'green'), pch=1)
```



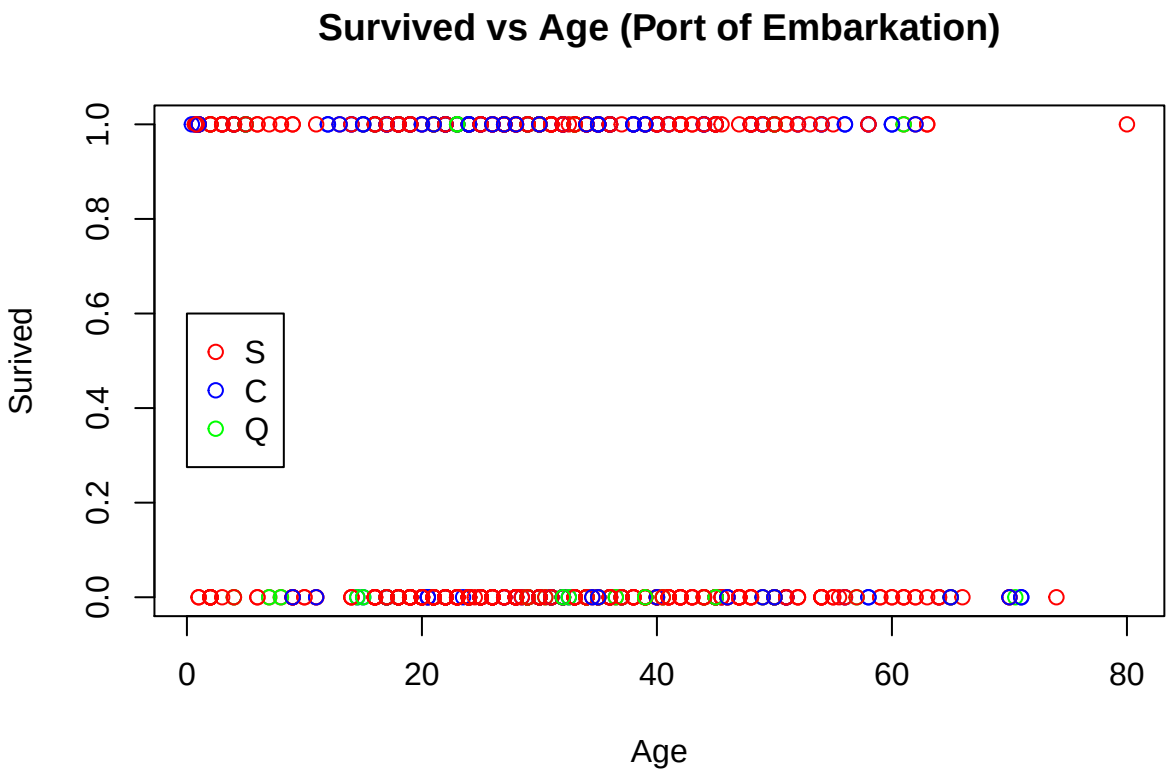
These plots are also insightful. The log transform appears to make a cleaner distribution of the Fare variable, and shows better seperation between the passenger classes. Accross almost all ages, the 3rd class passengers are much less likely to survive, while the 1st class customers are more likely to survive. The categorical Pclass variable will likely be impactful in my model.

I also want to see the impact of embarked on survival. Let's visualize once again.

```
emb_color <- ifelse(data_titanic$Embarked=='S', 'red', 'blue')
emb_color[data_titanic$Embarked=='Q'] <- 'green'

# survived vs fare
plot(log(data_titanic$Fare), data_titanic$Survived, col= emb_color,
     main = 'Survived vs log(Fare) (Port of Embarkation)',
     xlab = 'log(Ticket Price)', ylab='Surived')

legend(1.35, 0.6, legend = c('S', 'C', 'Q'), col = c('red', 'blue', 'green'), pch=1)
```



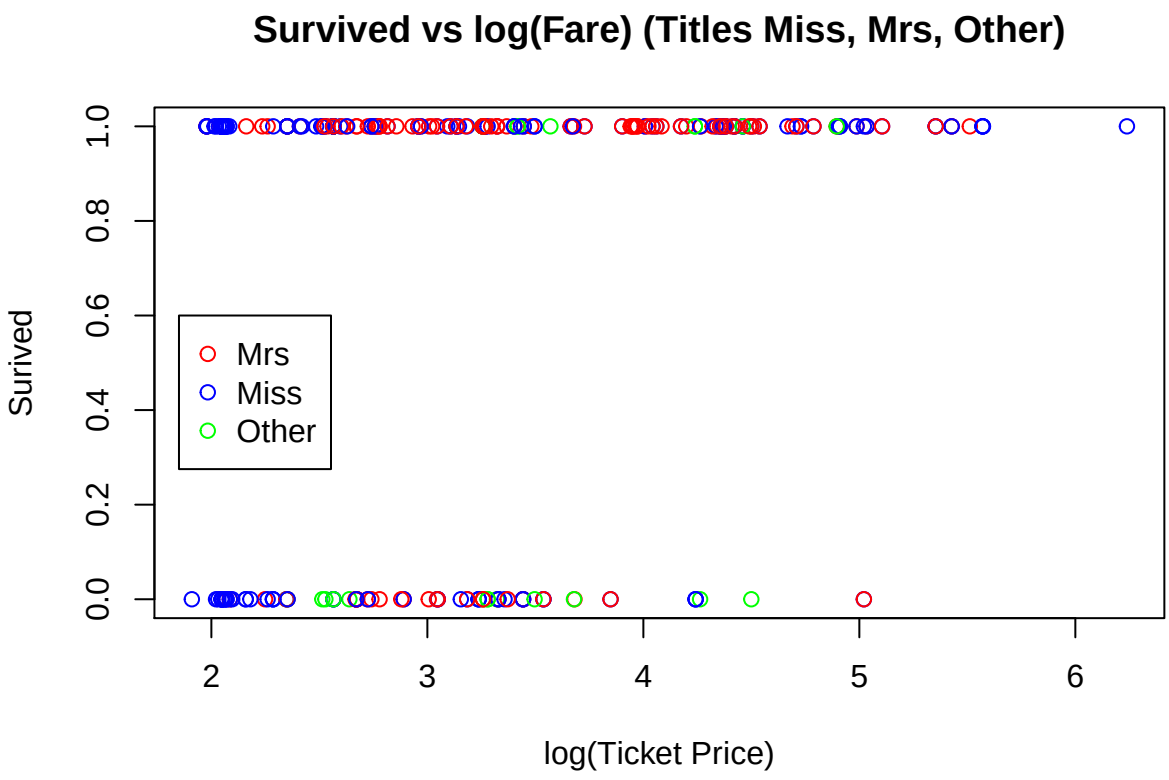

```
# survived vs age
plot(data_titanic$Age, data_titanic$Survived, col= emb_color,
     main='Survived vs Age (Port of Embarkation)', xlab = 'Age', ylab='Surived')
legend(0, 0.6, legend = c('S', 'C', 'Q'), col = c('red', 'blue', 'green'), pch=1)
```



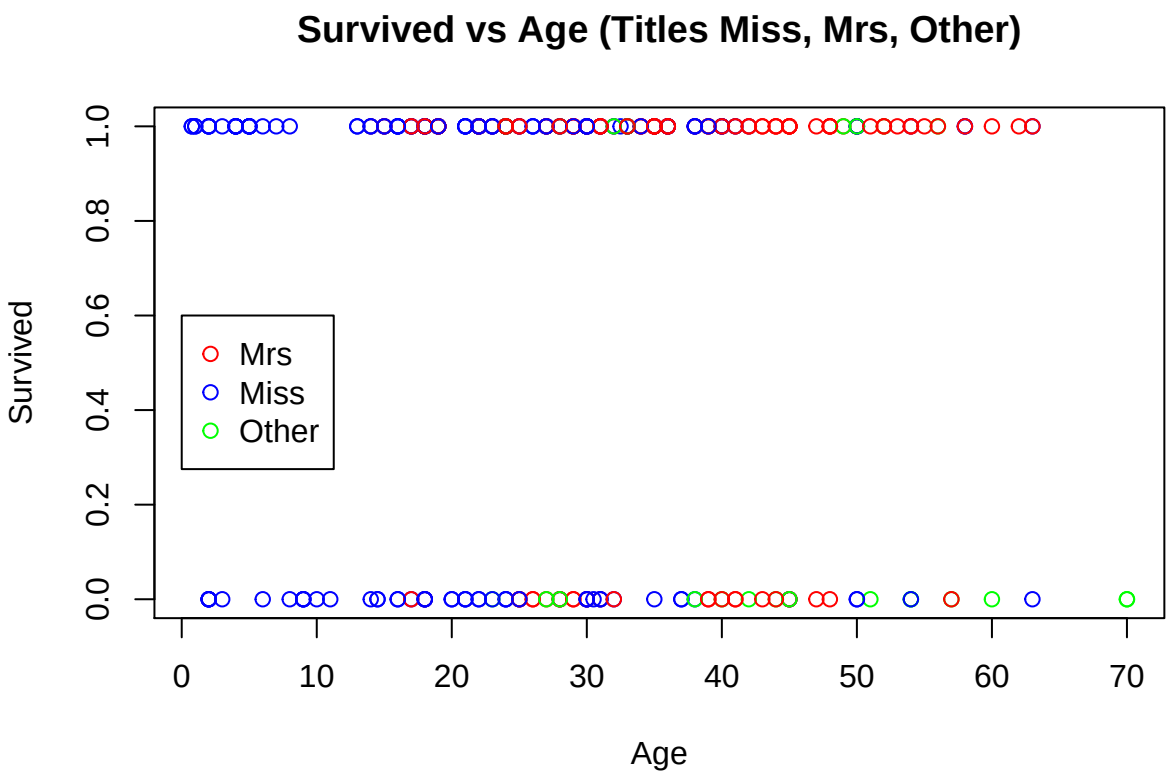
Embarkment port doesn't seem to have a significant impact on probability of surviving, however, passengers that embarked from Q (Queenstown) seemed to mostly not survive.

The last categorical variable of interest to me is the title variable. Since I know that passengers of the female sex survived, I want to see the difference between the different female titles but also include the other title.

```
title_color <- ifelse(data_titanic$Title=='Mrs', 'red', 'blue')
title_color[data_titanic$Title=='Other'] <- 'green'

title_color <-
  title_color[data_titanic$Title=='Mrs' | data_titanic$Title=='Miss' |
                              data_titanic$Title=='Other']

# survived vs fare
plot(log(data_titanic$Fare[data_titanic$Title == 'Miss' |
                              data_titanic$Title == 'Mrs' |
                              data_titanic$Title == 'Other']),
     data_titanic$Survived[data_titanic$Title == 'Miss' |
                              data_titanic$Title == 'Mrs' |
                              data_titanic$Title == 'Other'],
     col= title_color,
     main = 'Survived vs log(Fare) (Titles Miss, Mrs, Other)',
     xlab='log(Ticket Price)', ylab='Surived')

legend(1.85, 0.6, legend = c('Mrs', 'Miss', 'Other'),
       col = c('red', 'blue', 'green'), pch=1)
```




```
# survived vs age
plot(data_titanic$Age[data_titanic$Title == 'Miss' |
                      data_titanic$Title == 'Mrs' |
                      data_titanic$Title == 'Other'],
     data_titanic$Survived[data_titanic$Title == 'Miss' |
                           data_titanic$Title == 'Mrs' |
                           data_titanic$Title == 'Other'],
     col= title_color,
     main='Survived vs Age (Titles Miss, Mrs, Other)',
     xlab='Age', ylab='Survived')

legend(0, 0.6, legend = c('Mrs', 'Miss', 'Other'), col = c('red', 'blue', 'green'), pch=1)
```



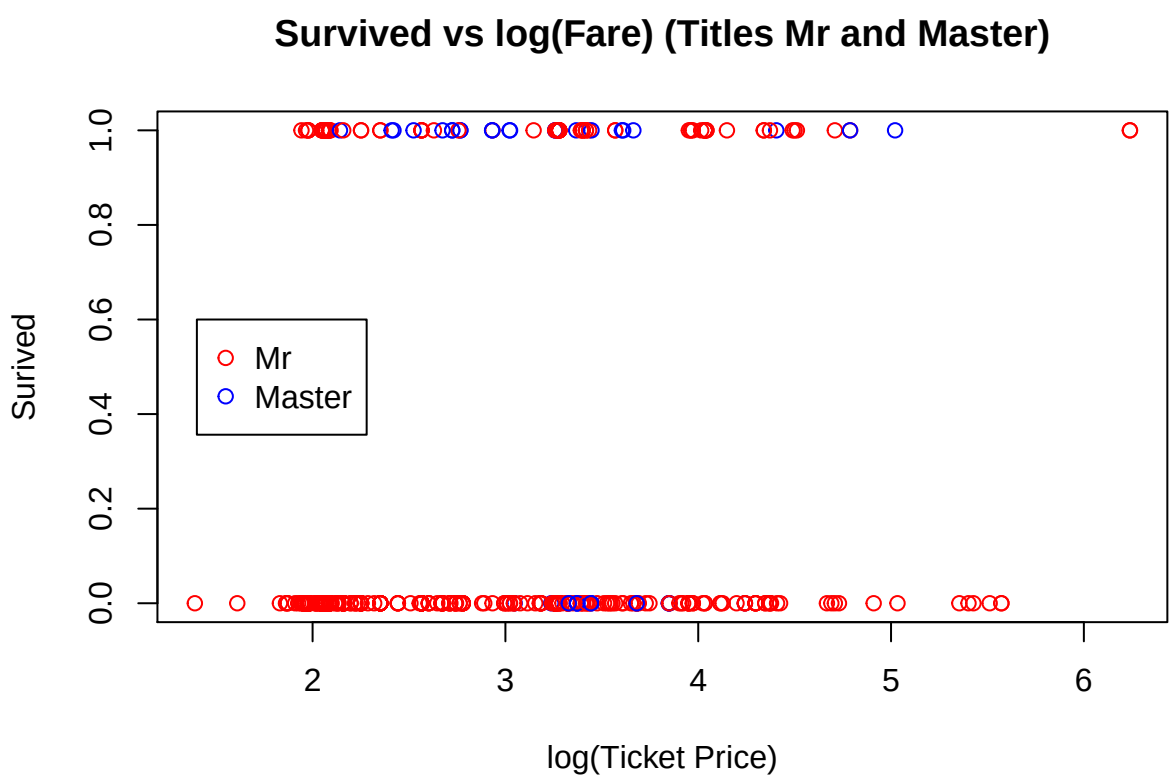
The plot above tells me that overall, passengers with the title Mrs had a much higher probability of survival accross all ages and ticket prices. There is not any other significant groupings to be seen in the plot, however, above the ticket price of about $e^4 \approx 55.5$, almost all of the passengers with this title survived.

These findings have made me curious to see if the titles Mr and Master (which are all male, as seen below), have any impact on probability of survival. The same visualization process will be repeated one last time.

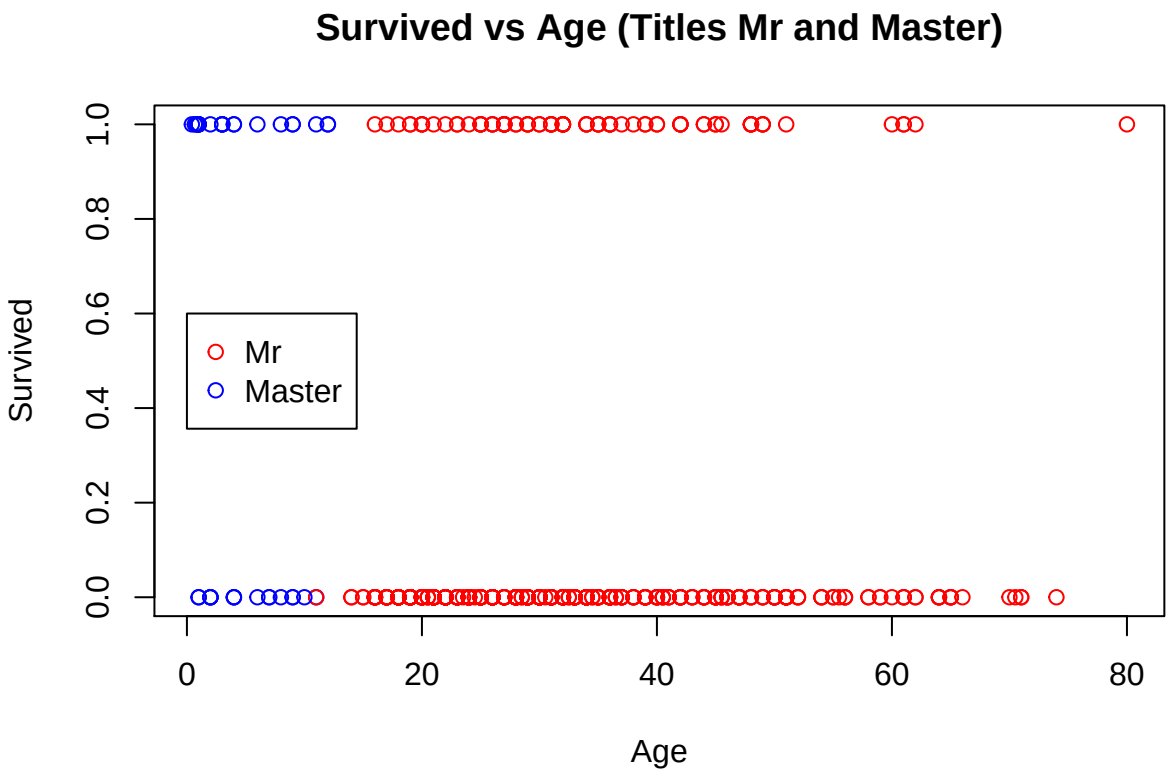
```
data_titanic$Sex[data_titanic$Title=='Master']
```

```
##  [1] "male" "male" "male" "male" "male" "male" "male" "male" "male" "male"
## [11] "male" "male" "male" "male" "male" "male" "male" "male" "male" "male"
## [21] "male" "male" "male" "male" "male" "male" "male" "male" "male" "male"
## [31] "male" "male" "male" "male" "male" "male" "male" "male" "male" "male"
```

```
title_color_MR_Master <- ifelse(data_titanic$Title=='Mr', 'red', 'blue')
title_color_MR_Master <- title_color_MR_Master[data_titanic$Title == 'Mr' |
                                               data_titanic$Title == 'Master']

# survived vs fare
plot(log(data_titanic$Fare[data_titanic$Title == 'Mr' | data_titanic$Title == 'Master']),
     data_titanic$Survived[data_titanic$Title == 'Mr' | data_titanic$Title == 'Master'],
     col= title_color_MR_Master, main = 'Survived vs log(Fare) (Titles Mr and Master)',
     xlab='log(Ticket Price)', ylab='Surived')
legend(1.40, 0.6, legend = c('Mr', 'Master'), col = c('red', 'blue'), pch=1)
```



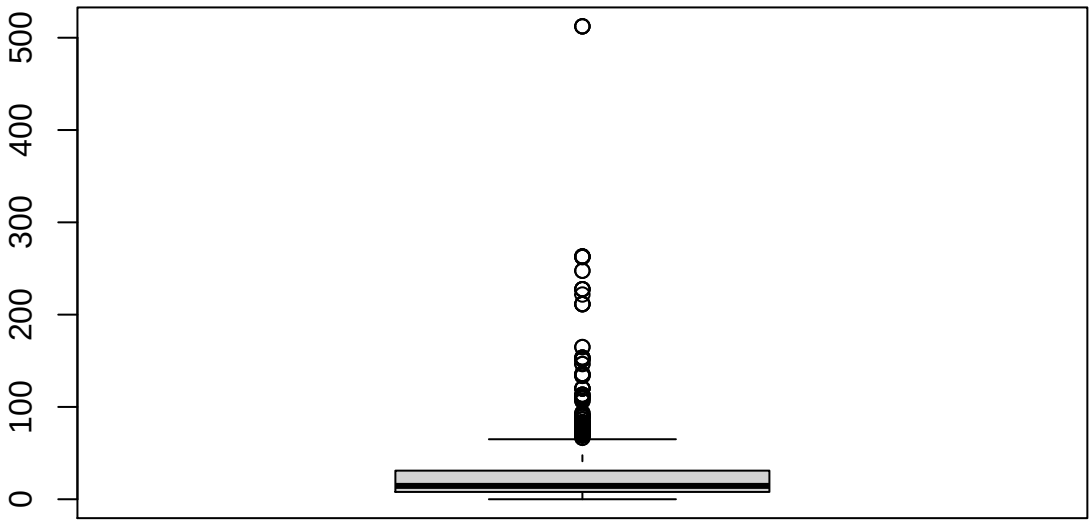

```
# survived vs age
plot(data_titanic$Age[data_titanic$Title == 'Mr' | data_titanic$Title == 'Master' ],
     data_titanic$Survived[data_titanic$Title == 'Mr' | data_titanic$Title == 'Master'],
     col= title_color_MR_Master, main='Survived vs Age (Titles Mr and Master)',
     xlab='Age', ylab='Survived')
legend(0, 0.6, legend = c('Mr', 'Master'), col = c('red', 'blue'), pch=1)
```



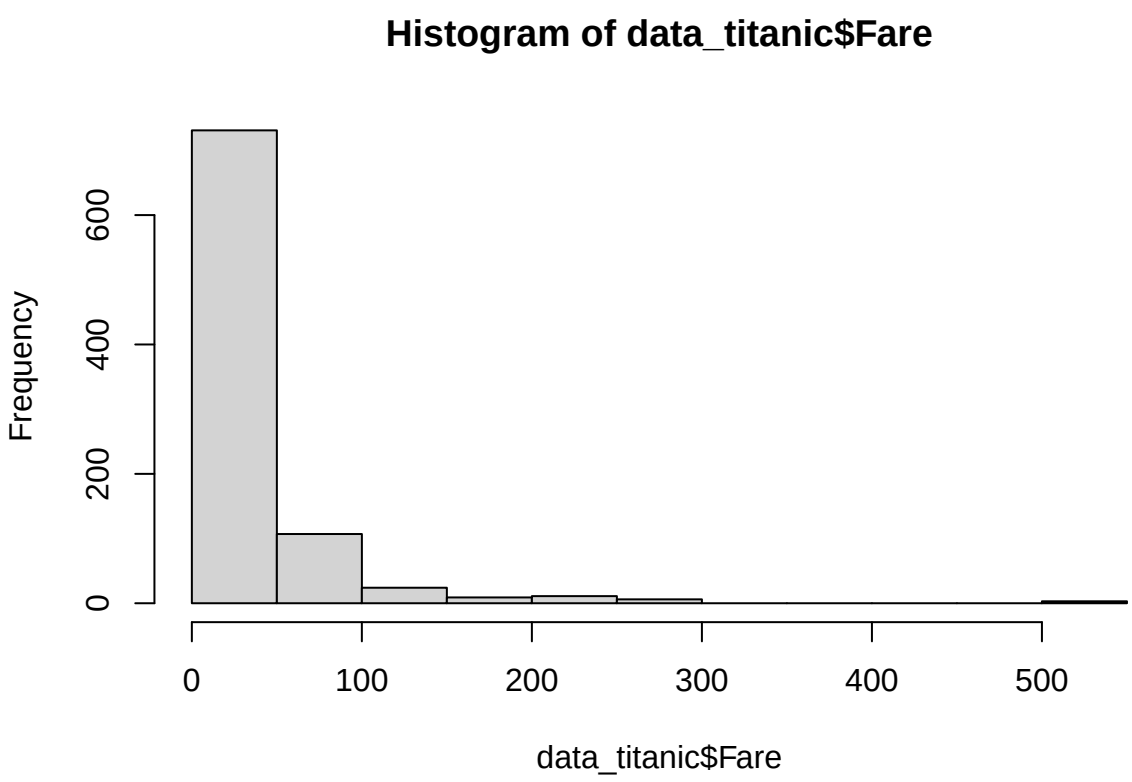
It appears that the fates of the Misters and the Masters are equally doomed to death, and I see no significance in the above plot. However, there be collinearity with Title and Age, as it appears the title Master is reserved for boys and not men.

Now that I better understand this data, I will take a look at the outliers.

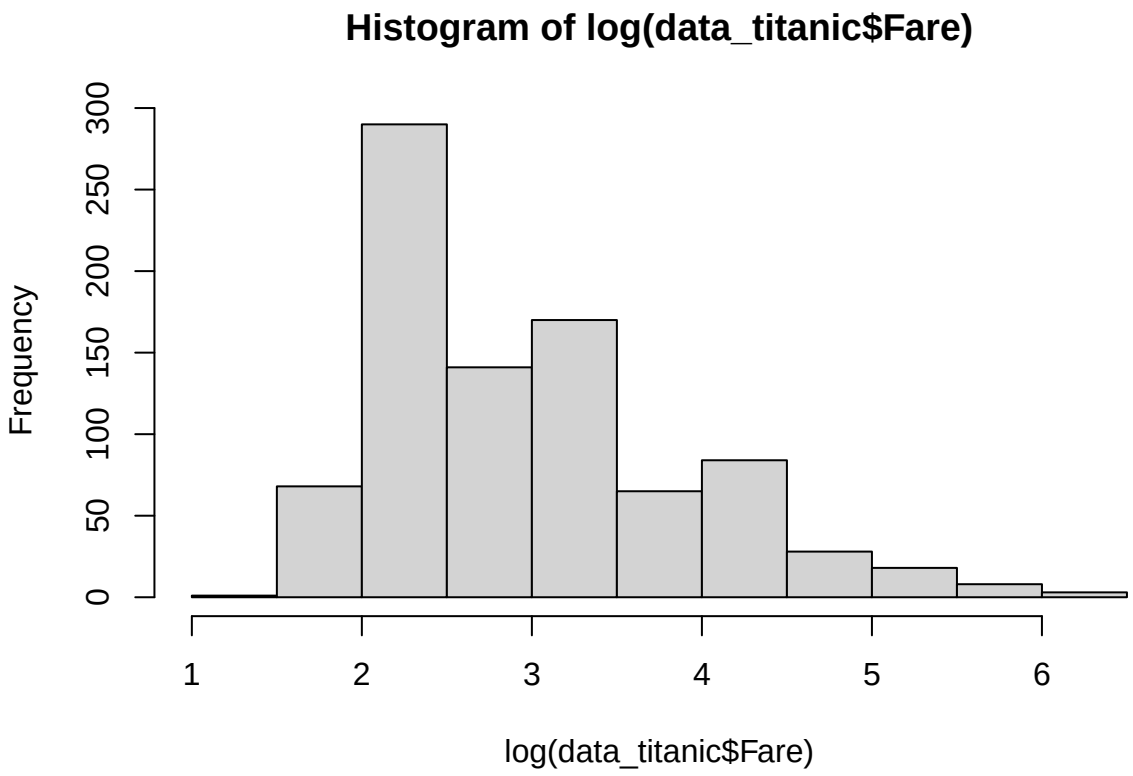
```
boxplot(data_titanic$Fare)
```




```r
hist(data_titanic$Fare)
```



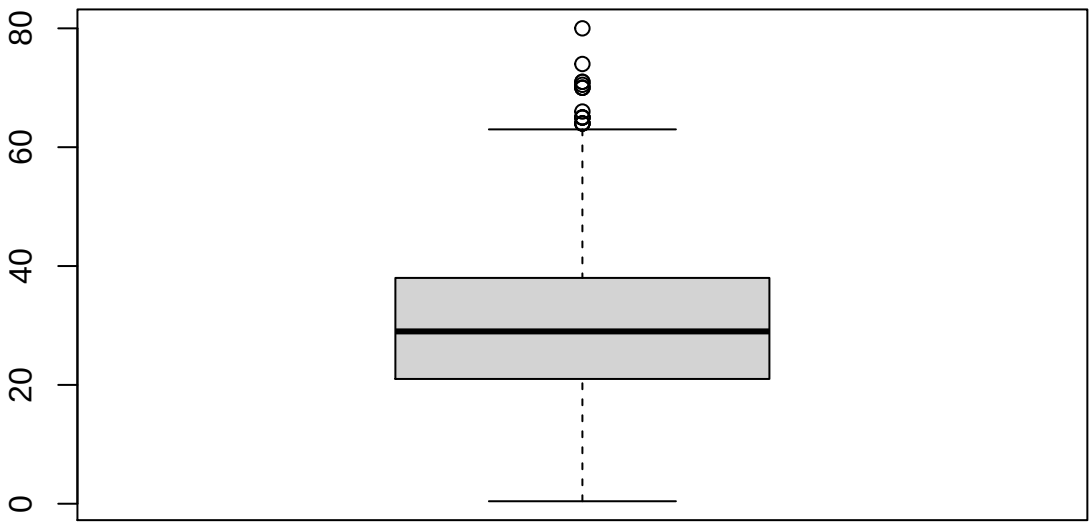

```r
hist(log(data_titanic$Fare))
```



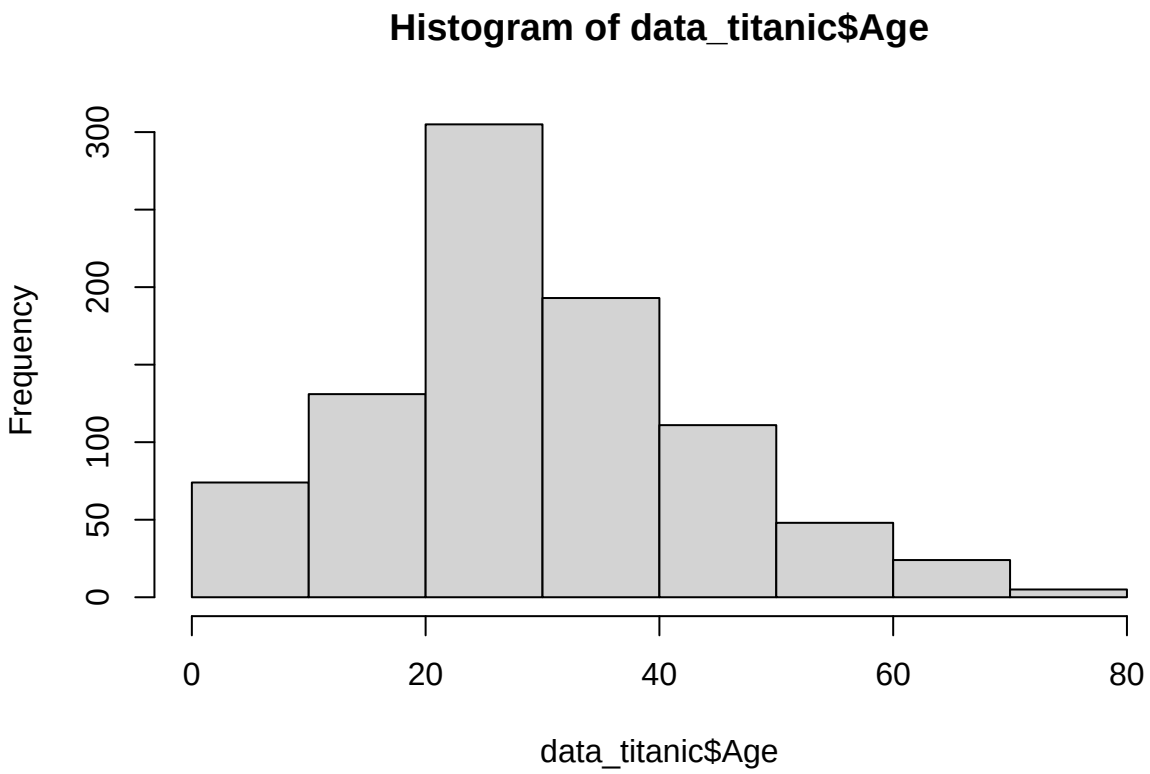
There are many outliers for Fare, with one in particular being even more extreme than the rest. Since there are many outliers and the histogram shows that there is not a normal distribution, I can't simply replace these outliers with the mean of the Fare variable. If I were to use this variable, I would more likely use to log transform (histogram shown) to reduce the impact of outliers.

Now looking at Age.

```
boxplot(data_titanic$Age)
```




```
hist(data_titanic$Age)
```



There are a few outliers for Age. The distirbution is near normal so it would be viable to replace the ages of the outliers with the mean instead of removing them. However, some of these Age values have been imputed, and further altering the values might create even more innacuracies from the true age distribution of the passengers. I will leave the Age variable untouched.

Model Seletion

Now, I will select the model I want to use. This dataset and the main task of creating a prediction rule for which passengers survive is a binary classification problem. I for this binary classification model, I will use a logistic (logit) model, which uses a binary response variable in a regression problem. This generalized linear model uses the logit link, which transforms our predictors and their regression coefficients to output a probability that the response variable (in this case survived) is equal to 0.

With this model selected, I can begin to build a training dataset. First I will need to remove variables that I know will be of no use, and convert appropriate variables to factors.

```
# keeping only the necessary columns
keeps <- c('Survived', 'Pclass', 'Sex', 'Age',
           'SibSp', 'Parch', 'Fare', 'Embarked', 'Title')
# making a cleaned dataset before I have a seperate frame for training
titanic_cleaned <- data_titanic[keeps]
names(titanic_cleaned)
```

```
## [1] "Survived" "Pclass"   "Sex"      "Age"      "SibSp"    "Parch"    "Fare"
## [8] "Embarked" "Title"
```

```
titanic_cleaned$Survived <- as.factor(titanic_cleaned$Survived)
titanic_cleaned$Sex <- as.factor(titanic_cleaned$Sex)
titanic_cleaned$Pclass <- as.factor(titanic_cleaned$Pclass)
titanic_cleaned$Embarked <- as.factor(titanic_cleaned$Embarked)
```

Despite being continous variables, Parch (parents and children onboard) and SibSp (siblings and spouses on board) do not have a high range of values and I think that leaving them as continous variables would not be appropriate. First let's see these ranges and what percent of passengers have the assigned values. Starting with SibSp.

```
# range of values
range(titanic_cleaned$SibSp)
```

```
## [1] 0 8
```

```
# proportion that have at least 1 sibling or spouse onboard
sum(titanic_cleaned$SibSp[titanic_cleaned$SibSp > 0]) / length(titanic_cleaned$SibSp)
```

```
## [1] 0.5230079
```

```
# proportion that have at least 5 siblings or spouse onboard
sum(titanic_cleaned$SibSp[titanic_cleaned$SibSp > 4]) / length(titanic_cleaned$SibSp)
```

```
## [1] 0.09090909
```

About half of the passengers have more than 1 sibling onboard. Perhaps a new feature that is a binary indicator for if the person has no siblings or spouses onboard would be more appropriate. Less than 10% of the passenger have 5 or more siblings onboard, but I think using this as an indicator as well would be useful since they represent large groups of people that are related onboard. Let's code these up before proceeding.

```
titanic_cleaned$NoSibSp <- ifelse(titanic_cleaned$SibSp == 0, 1, 0)
titanic_cleaned$NoSibSp <- as.factor(titanic_cleaned$NoSibSp)

titanic_cleaned$FiveOrMoreSibSp <- ifelse(titanic_cleaned$SibSp > 4, 1, 0)
titanic_cleaned$FiveOrMoreSibSp <- as.factor(titanic_cleaned$FiveOrMoreSibSp)
```

Now looking at the Parch (parents and children onboard).

```
range(titanic_cleaned$Parch)
```

```
## [1] 0 6
```

```
#proportion of passengers with at least 1 parent or child
sum(titanic_cleaned$Parch[titanic_cleaned$Parch > 0]) / nrow(titanic_cleaned)
```

```
## [1] 0.3815937
```

```r
# proportion of passengers with 4 or more passengers or children
sum(titanic_cleaned$Parch[titanic_cleaned$Parch > 3]) / nrow(titanic_cleaned)
```

```
## [1] 0.05274972
```

With only about 38% of the passengers having at least 1 parent or children on board, I think a single dummy variable to indicate the presence or absence of of parents and siblings onboard will be suitable. Let's code this as well, by changing the given column into a column binary indicator variable.

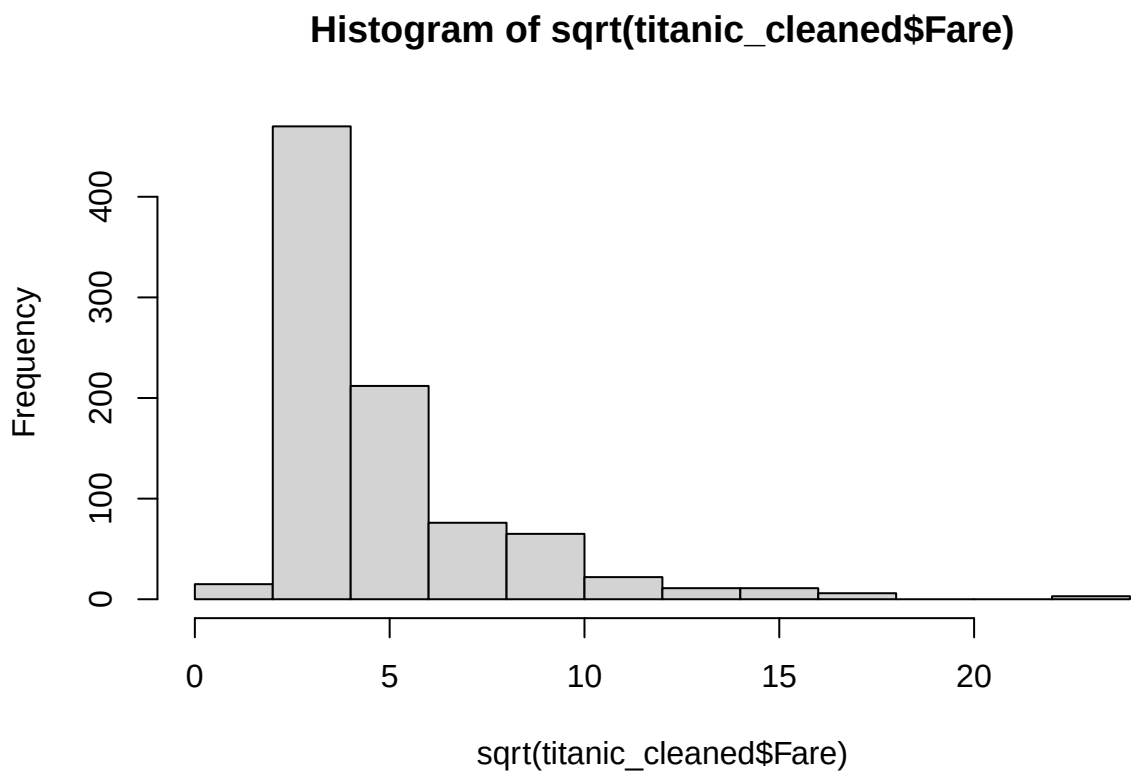
```r
titanic_cleaned$Parch <- ifelse(titanic_cleaned$Parch > 0, 1, 0)
titanic_cleaned$Parch <- as.factor(titanic_cleaned$Parch)
```

Now I must decide how to handle the Title variable. I am not sure what exactly to do with this variable. One possible idea is to soley have indicators for Miss and Mrs, however, this will likely result in collinearity with the Sex variable. Another idea is to have an indicator just for those with the title Mrs, since they had the highest probability of survival. For now, I am going to leave this variable untouched.

```r
titanic_cleaned$Title <- as.factor(titanic_cleaned$Title)
```

Lastly, the log transform for the Fare variable (ticket price). The justifications for this transform were demonstrated earlier, where its distribution became more normal and it significantly decreased the variance in in Fare values, bringing the outliers closer to the rest of the ticket prices. However, I have discovered that the variable has a range of 0-512.3292. This means that a square root transform will need to be check for instead.

```r
hist(sqrt(titanic_cleaned$Fare))
```



This distribution is not quite normal. However, I think that the variable is to important to drop, and I will remove it later if it is causing issues in model performance.

```
titanic_cleaned$sqrtFare <- sqrt(titanic_cleaned$Fare)
# dropping Fare variable
titanic_cleaned<- titanic_cleaned[, -which(names(titanic_cleaned) %in% c("Fare"))]
names(titanic_cleaned)
```

```
##  [1] "Survived"        "Pclass"          "Sex"             "Age"
##  [5] "SibSp"           "Parch"           "Embarked"        "Title"
##  [9] "NoSibSp"         "FiveOrMoreSibSp" "sqrtFare"
```

Before building and applying a training and validation split to the dataset, I will be using Variable Inflation Factor (VIF, https://www.statisticshowto.com/variance-inflation-factor/) to check for multi-collinearity amongst the variables. VIF values of 1 indicate no collinearity, less than 5 indicate some degree of collinearity, and 5 or greater is a signficant amount of collinearity.

```
# using a GLM on the whole dataset to look for multicollinearity
# before proceeding to training model on train split of dataset
titanic_cleaned$OneToFourSibSp
```

```
## NULL
```

```
library(car)
```

```
## Warning: package 'car' was built under R version 4.0.3
```

```
## Loading required package: carData
```

```
## Warning: package 'carData' was built under R version 4.0.3
```

```
vif(glm(Survived~.-SibSp, data=titanic_cleaned, family='binomial'))
```

```
##                         GVIF Df GVIF^(1/(2*Df))
## Pclass          2.660134e+00  2        1.277103
## Sex             3.017512e+07  1     5493.188458
## Age             1.842041e+00  1        1.357218
## Parch           1.817375e+00  1        1.348100
## Embarked        1.303097e+00  3        1.045112
## Title           6.815212e+07  4        9.532019
## NoSibSp         1.418020e+00  1        1.190807
## FiveOrMoreSibSp 1.000000e+00  1        1.000000
## sqrtFare        2.254656e+00  1        1.501551
```

As I suspected, the titles are collinear with Sex and the Sex variable has an astranomically high VIF value. Let's see if removing it solves the value for Title that is currently greater than 5 (and we want to decrease).

```
vif(glm(Survived~.-SibSp-Sex, data=titanic_cleaned, family='binomial'))
```

```
##                     GVIF Df GVIF^(1/(2*Df))
## Pclass          2.705956  2        1.282567
## Age             1.829446  1        1.352570
## Parch           1.825284  1        1.351031
## Embarked        1.304504  3        1.045300
## Title           2.823880  4        1.138560
## NoSibSp         1.425723  1        1.194036
## FiveOrMoreSibSp 1.000000  1        1.000000
## sqrtFare        2.255769  1        1.501922
```

Removing Sex from the VIF calculation has solved all isues of collinearity, and in fact, the highest VIF score is only 1.502 which is excellent. I will remove Sex from my calculation, split my cleaned dataset into train and validte sections, then start training a model.

```
# removing Sex variable
titanic_cleaned <- titanic_cleaned[, -which(names(titanic_cleaned) %in% c("Sex"))]
```

I have decided to use a 85/15 ratio for train / validate for my model. The current data I have been working with is 68% of the total data (where the testing dataset is 32%). Even at an 85/25 split, I am training on about 58% of the data which is suboptimal in my opinion.

```
set.seed(12345)
tr <- sample(1:nrow(titanic_cleaned), 0.85 * nrow(titanic_cleaned))
titanic_train <- titanic_cleaned[tr,]
titanic_val <- titanic_cleaned[-tr,]
```

Now I can build my logit model, starting with all of the variables in my training dataset. Just to note, this data is my already cleaned data with certain features engineered for the training. Also, I have kept the original siblings and parents variable (SibSp) in case my dummy variables for the data does not yield favorable results; it will be left out of the model to begin with.

```
model.titanic1 <- glm(Survived~.-SibSp, data=titanic_train, family='binomial')
summary(model.titanic1)
```

```
##
## Call:
## glm(formula = Survived ~ . - SibSp, family = "binomial", data = titanic_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2807  -0.6221  -0.3706   0.6142   2.4690
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)       1.609e+01  1.696e+03   0.009 0.992428
## Pclass2          -9.598e-01  3.564e-01  -2.693 0.007085 **
## Pclass3          -2.016e+00  3.619e-01  -5.572 2.52e-08 ***
## Age              -1.471e-02  8.633e-03  -1.704 0.088455 .
## Parch1           -6.005e-01  2.960e-01  -2.029 0.042469 *
## EmbarkedC        -1.324e+01  1.696e+03  -0.008 0.993769
## EmbarkedQ        -1.349e+01  1.696e+03  -0.008 0.993651
## EmbarkedS        -1.376e+01  1.696e+03  -0.008 0.993527
```

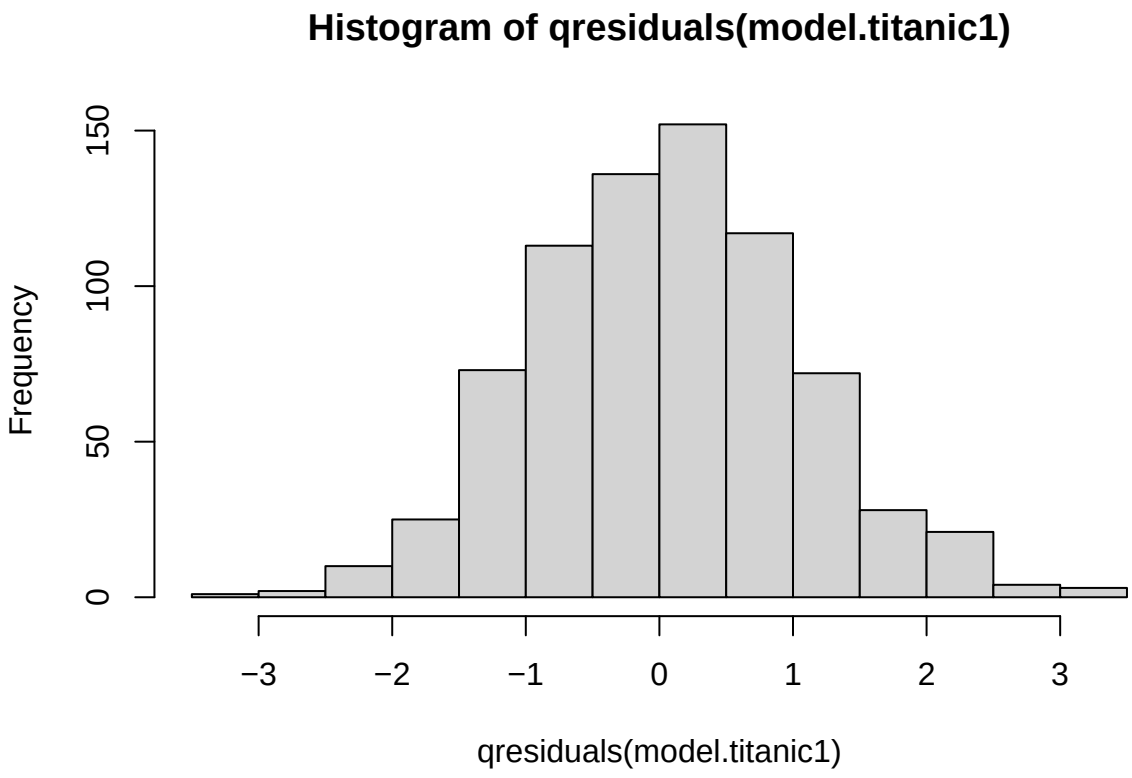
```
## TitleMiss        -1.385e-01  4.957e-01  -0.279 0.779950
## TitleMr          -3.137e+00  5.539e-01  -5.664 1.48e-08 ***
## TitleMrs          3.371e-01  5.516e-01   0.611 0.541085
## TitleOther       -2.790e+00  7.635e-01  -3.654 0.000258 ***
## NoSibSp1          5.157e-01  2.515e-01   2.051 0.040280 *
## FiveOrMoreSibSp1 -1.589e+01  6.460e+02  -0.025 0.980380
## sqrtFare          6.647e-02  5.208e-02   1.276 0.201866
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1010.46  on 756  degrees of freedom
## Residual deviance:  642.95  on 742  degrees of freedom
## AIC: 672.95
##
## Number of Fisher Scoring iterations: 15
```

Checking residuals

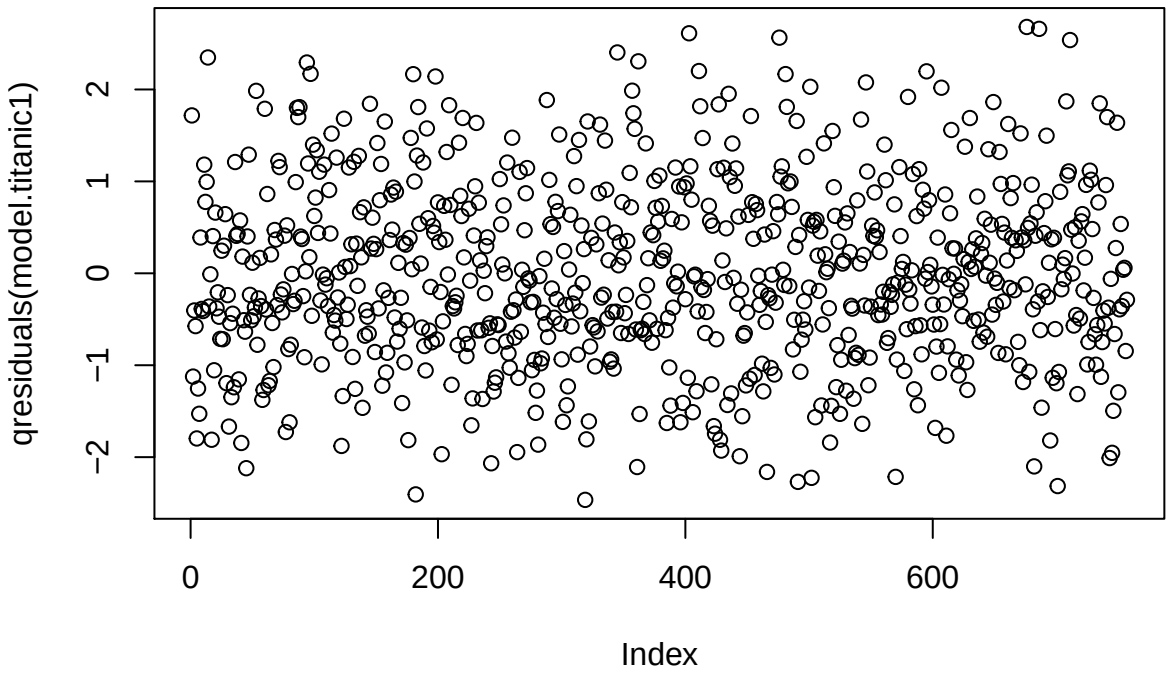
```
library(statmod)
```

```
## Warning: package 'statmod' was built under R version 4.0.3
```

```
hist(qresiduals(model.titanic1))
```



```
plot(qresiduals(model.titanic1))
```



There are not many significant terms in the summary of the model. However, the residuals are normal and I think some adjustments could perhaps improve the model. First lets test the model on the validation data. I have found an analog of RMSE for binary classification problems that is called the Brier score (https://en.wikipedia.org/wiki/Brier_score), which I found from the post here: https://stats.stackexchange.com/a/172985. I will be using it to get a better assessment of out of sample performance for my models than simply an accuracy would. As the stackexchange post discuses, simply using accuracy is not the best assessment. This is because I might not have an ideal threshold for probabilities to be classified as survived or not survived (and this issue that I was pondering is what led me to look for a better assessment criteria). My Brier score is obtained with the equation $BS = \frac{1}{N}\sum_{t=1}^{N}(P_t - o_t)^2$, where N is the length of my validation set, P_t is my predicted probabilities for passenger survival, and o_t is the actual outcome for the passenger. It is the mean squared differnce between predicted probability of survival and the true outcome, a suitable analog. A smaller Brier score, among other criteria, will help me determine which model is best. I will still use accuracy to assess performance, as well as the F1 score (harmonic mean of recall=True Positive Rate and precision=proportion of positive values predicted correctly), and specificty= True Negative Rate

I am going to build a dataframe to keep track of these scores.

```
performance <- data.frame(modelName = 'model.titanic1, no interacitons',
                          BICscore = extractAIC(model.titanic1,
                                                k=log(nrow(titanic_train)))[2])
```

```
model.titanic.pred1 <- predict(model.titanic1,
                               newdata=titanic_val[,2:length(titanic_train)],
```

```
                                type='response')
# Brier score for the model
# getting outcome values for brier score (need to make the survived factor a numeric,
# and subtract 1 since an addition of 1 occurs in the conversion)
outcomes_val <- as.numeric(titanic_val[,1]) - 1
# computing Brier score and adding to my dataframe
performance$BrierScore <-
  (1/nrow(titanic_val)) * sum((model.titanic.pred1 - outcomes_val)^2)
```

computing accuracy, recall and f1 score

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.0.3
```

```
## Loading required package: lattice
```

```
#re-leveling Survived for the confusion matrix function
titanic_val$Survived <- relevel(titanic_val$Survived, '1')
# providing predictions converted to outcomes using threshold of 0.5
CM1 <- confusionMatrix(data = relevel(as.factor(ifelse(model.titanic.pred1 >0.5, 1, 0)),
                                      '1'),
                       reference= titanic_val$Survived, mode='everything')
# confusion matrix
CM1$table
```

```
##           Reference
## Prediction  1  0
##          1 38 10
##          0 11 75
```

```
# storing values
performance$accuracy <- CM1$overall[[1]]
performance$F1 <- CM1$byClass[[7]]
performance$specificity = CM1$byClass[[2]]

# viewing metrics relating to confusion matrix
CM1
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  0
##          1 38 10
##          0 11 75
##
##                Accuracy : 0.8433
##                  95% CI : (0.7705, 0.9003)
##     No Information Rate : 0.6343
```

```
##     P-Value [Acc > NIR] : 7.786e-08
##
##                   Kappa : 0.6607
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.7755
##             Specificity : 0.8824
##          Pos Pred Value : 0.7917
##          Neg Pred Value : 0.8721
##               Precision : 0.7917
##                  Recall : 0.7755
##                      F1 : 0.7835
##              Prevalence : 0.3657
##          Detection Rate : 0.2836
##    Detection Prevalence : 0.3582
##       Balanced Accuracy : 0.8289
##
##        'Positive' Class : 1
##
```

```
# performance on first model
performance
```

```
##                         modelName BICscore BrierScore  accuracy        F1
## 1 model.titanic1, no interacitons 742.3893  0.1244595 0.8432836 0.7835052
##   specificity
## 1   0.8823529
```

Now I will create a new model using the step function and repeat this process. For this BIC step selected model to be 'approved', it should have a higher BIC probability than my previous model. Also, it will need to improve accuracy and F1 as a main criteria.

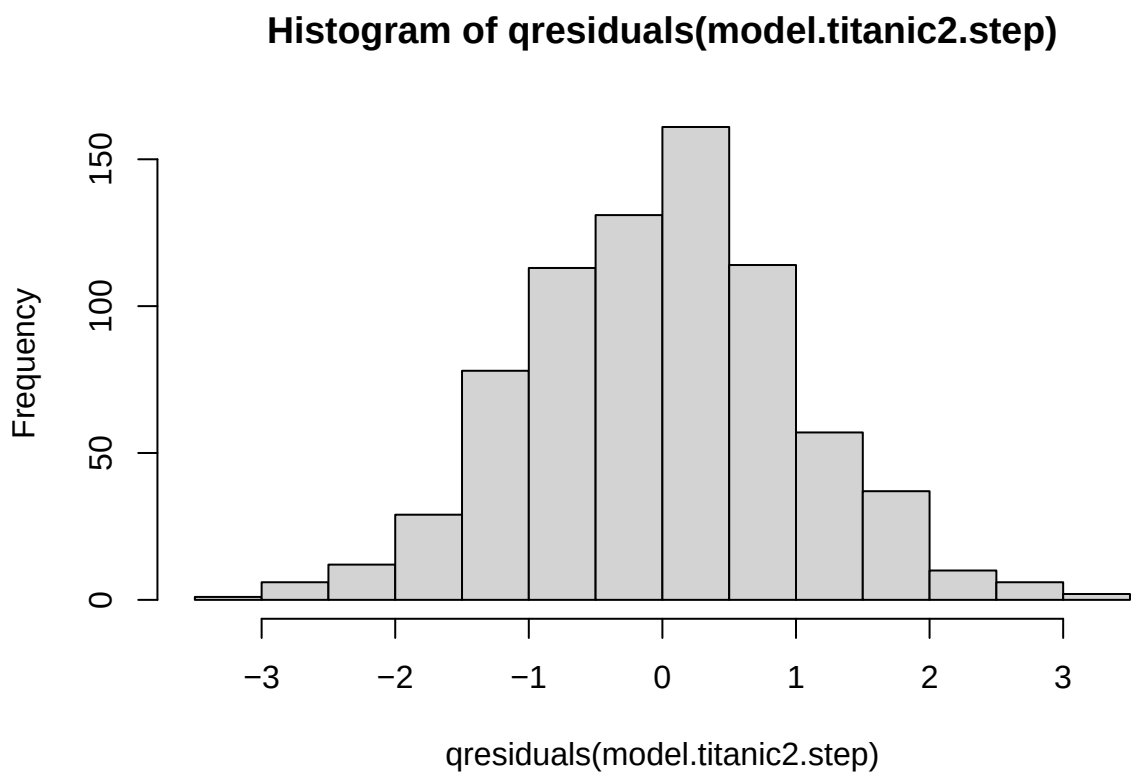
```
null <- glm(Survived ~1, data=titanic_train, family='binomial')
full <- glm(Survived~.-SibSp, data=titanic_train, family='binomial')
model.titanic2.step <- step(object = null, scope=formula(full),
                            direction='forward', k = log(nrow(titanic_train)), trace=F)
summary(model.titanic2.step)
```

```
##
## Call:
## glm(formula = Survived ~ Title + Pclass + FiveOrMoreSibSp, family = "binomial",
##     data = titanic_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3287  -0.6186  -0.3868   0.6428   2.2935
##
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)        2.3140     0.4600   5.030 4.90e-07 ***
## TitleMiss          0.2911     0.4424   0.658 0.510438
## TitleMr           -2.6997     0.4350  -6.206 5.44e-10 ***
```

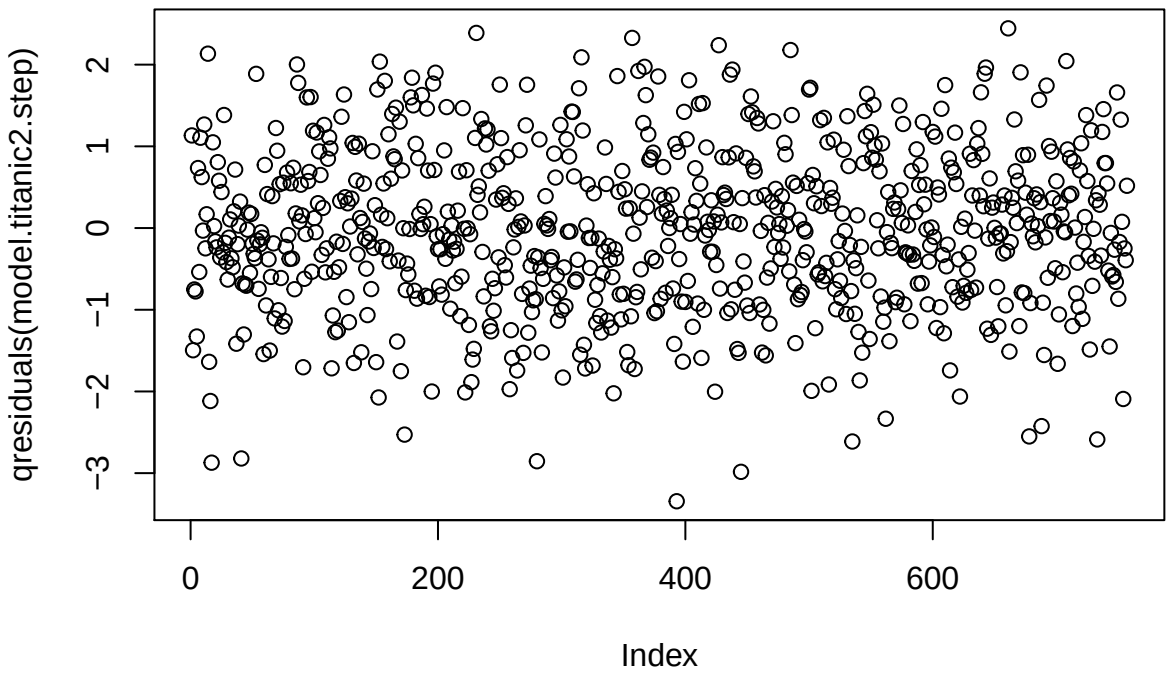
```
## TitleMrs           0.3286     0.4722   0.696 0.486466
## TitleOther        -2.4062     0.6453  -3.729 0.000192 ***
## Pclass2           -1.1708     0.2838  -4.125 3.70e-05 ***
## Pclass3           -2.1696     0.2546  -8.522  < 2e-16 ***
## FiveOrMoreSibSp1 -16.2169   654.1155  -0.025 0.980221
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1010.46  on 756  degrees of freedom
## Residual deviance:  661.31  on 749  degrees of freedom
## AIC: 677.31
##
## Number of Fisher Scoring iterations: 15
```

Residuals:

```
hist(qresiduals(model.titanic2.step))
```




```
plot(qresiduals(model.titanic2.step))
```



The model, seems to be improved, as there are more significant terms in the summary output. Also, the intercept, which includes the first levels of the created dummy variables for features with more than 1 category (like ticket class), is not significant. The residuals once again satisfy the regression assumptions with a normal distribution centered on 0 and a constant variance.

Now, we will collect all of the necessary metrics for out of sample predictive performance.

```
# adding new row to the performance dataframe
performance[2,] <- NA
performance$modelName[2] <- 'model.titanic2.step, no interactions'
performance$BICscore[2] <-  extractAIC(model.titanic2.step, k=log(nrow(titanic_train)))[2]

# prediction
model.titanic.pred2 <- predict(model.titanic2.step,
                               newdata=titanic_val[,2:length(titanic_train)],
                               type='response')

# computing Brier score and adding to my dataframe
performance$BrierScore[2] <-
  (1/nrow(titanic_val)) * sum((model.titanic.pred2 - outcomes_val)^2)

# providing predictions converted to outcomes using threshold of 0.5
CM2 <- confusionMatrix(data = relevel(as.factor(ifelse(model.titanic.pred2 >0.5, 1, 0)),
                                      '1'),
                       reference= titanic_val$Survived, mode='everything')

# confusion matrix
```

```
CM2$table
```

```
##           Reference
## Prediction  1  0
##          1 38 15
##          0 11 70
```

```
# storing values
performance$accuracy[2] <- CM2$overall[[1]]
performance$F1[2] <- CM2$byClass[[7]]
performance$specificity[2] <- CM2$byClass[[2]]

# viewing metrics relating to confusion matrix
CM2
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  0
##          1 38 15
##          0 11 70
##
##                Accuracy : 0.806
##                  95% CI : (0.7288, 0.8692)
##     No Information Rate : 0.6343
##     P-Value [Acc > NIR] : 1.216e-05
##
##                   Kappa : 0.5889
##
##  Mcnemar's Test P-Value : 0.5563
##
##             Sensitivity : 0.7755
##             Specificity : 0.8235
##          Pos Pred Value : 0.7170
##          Neg Pred Value : 0.8642
##               Precision : 0.7170
##                  Recall : 0.7755
##                      F1 : 0.7451
##              Prevalence : 0.3657
##          Detection Rate : 0.2836
##    Detection Prevalence : 0.3955
##       Balanced Accuracy : 0.7995
##
##        'Positive' Class : 1
##
```

```
# performance on models
performance
```

```
##                             modelName BICscore BrierScore  accuracy        F1
## 1      model.titanic1, no interacitons 742.3893  0.1244595 0.8432836 0.7835052
## 2 model.titanic2.step, no interactions 714.3441  0.1328708 0.8059701 0.7450980
```

```
##   specificity
## 1   0.8823529
## 2   0.8235294
```

The difference in out of sample performance on my validation set is interesting. While BIC score has decreased (like due largely to decreased model complexity), the Brier Score has slightly increased (which is worse performance), my accuracy, F1, and specificity have decreased quite a bit. Overall, I would say that my BIC step selected model is worse, which I find very interesting. These results encourage me to try one more model before looking at interaction models. I will create a new model that adjusts my first model.

```
summary(model.titanic1)
```

```
##
## Call:
## glm(formula = Survived ~ . - SibSp, family = "binomial", data = titanic_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2807  -0.6221  -0.3706   0.6142   2.4690
##
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)       1.609e+01  1.696e+03   0.009 0.992428
## Pclass2          -9.598e-01  3.564e-01  -2.693 0.007085 **
## Pclass3          -2.016e+00  3.619e-01  -5.572 2.52e-08 ***
## Age              -1.471e-02  8.633e-03  -1.704 0.088455 .
## Parch1           -6.005e-01  2.960e-01  -2.029 0.042469 *
## EmbarkedC        -1.324e+01  1.696e+03  -0.008 0.993769
## EmbarkedQ        -1.349e+01  1.696e+03  -0.008 0.993651
## EmbarkedS        -1.376e+01  1.696e+03  -0.008 0.993527
## TitleMiss        -1.385e-01  4.957e-01  -0.279 0.779950
## TitleMr          -3.137e+00  5.539e-01  -5.664 1.48e-08 ***
## TitleMrs          3.371e-01  5.516e-01   0.611 0.541085
## TitleOther       -2.790e+00  7.635e-01  -3.654 0.000258 ***
## NoSibSp1          5.157e-01  2.515e-01   2.051 0.040280 *
## FiveOrMoreSibSp1 -1.589e+01  6.460e+02  -0.025 0.980380
## sqrtFare          6.647e-02  5.208e-02   1.276 0.201866
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1010.46  on 756  degrees of freedom
## Residual deviance:  642.95  on 742  degrees of freedom
## AIC: 672.95
##
## Number of Fisher Scoring iterations: 15
```

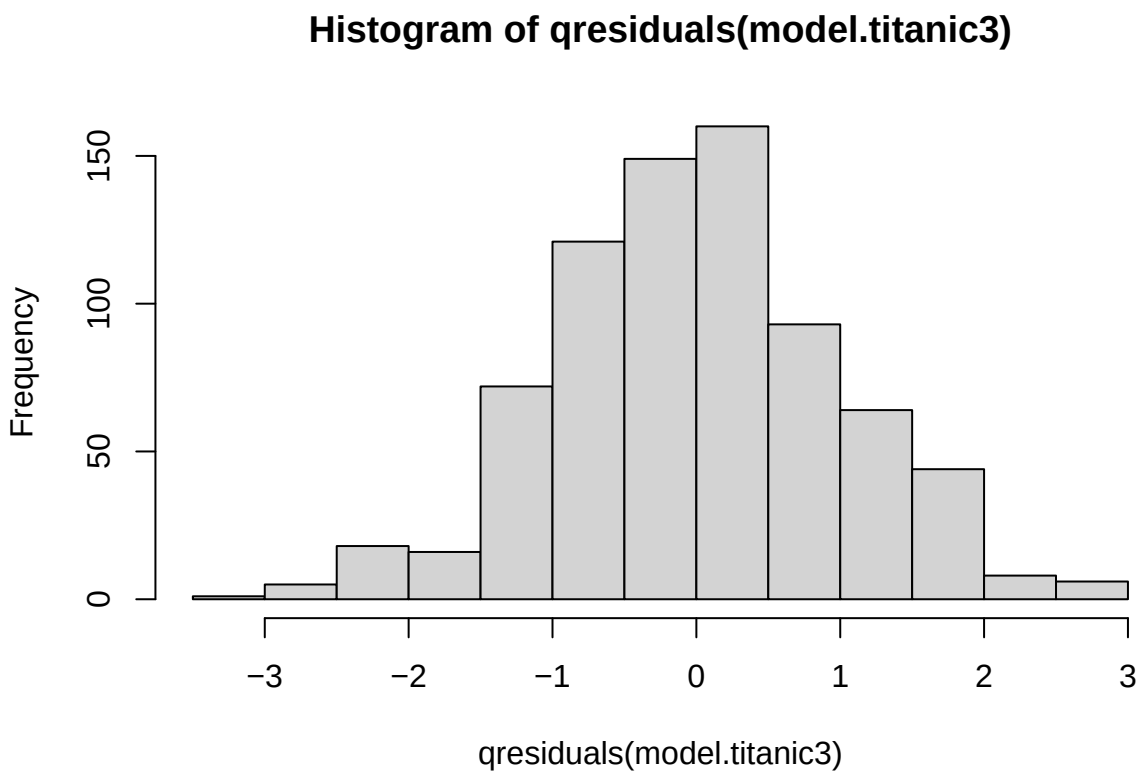
I believe removing the Embark variable and the feature I created for having 5 or more siblings or spouses onboard should be removed. The Embark variables appear to be near useless (and my earlier visualizations indicated this). The feature for 5 or more siblings or spouses also has a very high p-value, and appears to be unecessary. I will leave in the rest of the variables from the model.

```
model.titanic3 <- glm(Survived~. -SibSp - FiveOrMoreSibSp - Embarked, data=titanic_train,
                      family='binomial')
summary(model.titanic3)
```

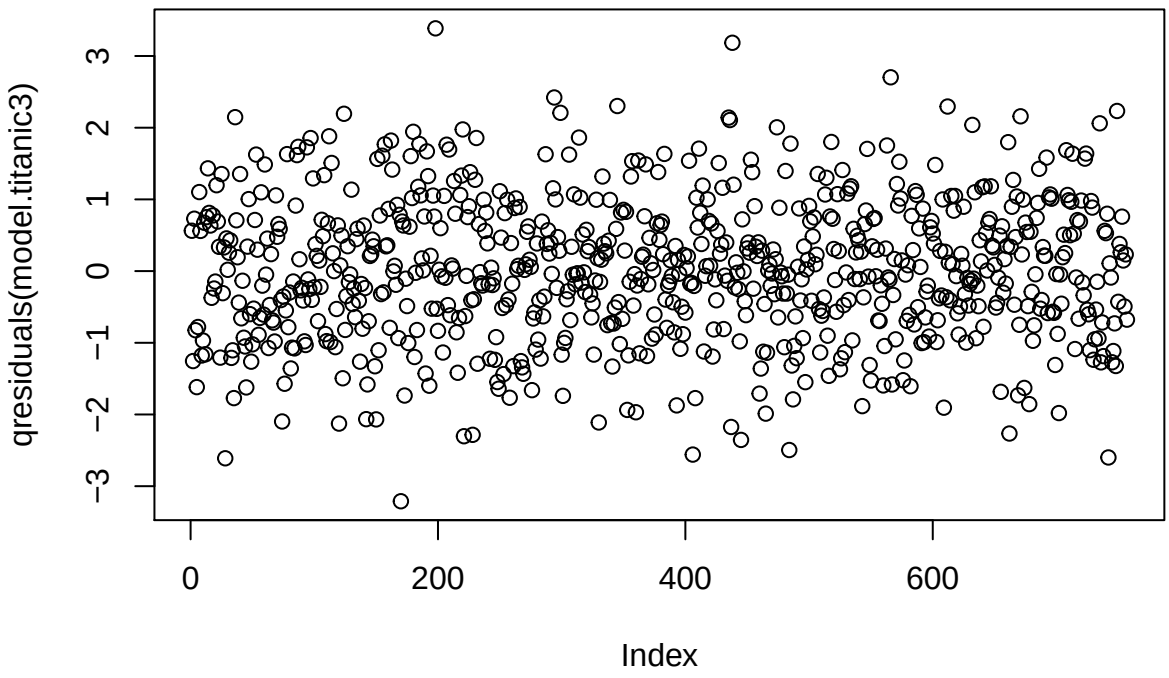
```
##
## Call:
## glm(formula = Survived ~ . - SibSp - FiveOrMoreSibSp - Embarked,
##     family = "binomial", data = titanic_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3553  -0.6465  -0.3780   0.5985   2.6435
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  2.334821   0.674396   3.462 0.000536 ***
## Pclass2     -1.135957   0.348951  -3.255 0.001133 **
## Pclass3     -2.161785   0.348669  -6.200 5.64e-10 ***
## Age         -0.016647   0.008568  -1.943 0.052038 .
## Parch1      -0.660177   0.290640  -2.271 0.023119 *
## TitleMiss    0.124850   0.464321   0.269 0.788016
## TitleMr     -2.899829   0.519564  -5.581 2.39e-08 ***
## TitleMrs     0.654054   0.521538   1.254 0.209810
## TitleOther  -2.543876   0.732997  -3.471 0.000519 ***
## NoSibSp1     0.591408   0.248959   2.376 0.017524 *
## sqrtFare     0.064726   0.049457   1.309 0.190618
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1010.46  on 756  degrees of freedom
## Residual deviance:  656.89  on 746  degrees of freedom
## AIC: 678.89
##
## Number of Fisher Scoring iterations: 5
```

Checking residuals

```
hist(qresiduals(model.titanic3))
```




```
plot(qresiduals(model.titanic3))
```



My residual assumptions are satisfied. This model has improved on the first models number of significant terms. Let's see if the out of sample performance has improved as well.

```
# adding new row to the performance dataframe
performance[3,] <- NA
performance$modelName[3] <- 'model.titanic3, no interactions'
performance$BICscore[3] <-  extractAIC(model.titanic3, k=log(nrow(titanic_train)))[2]

# prediction
model.titanic.pred3 <- predict(model.titanic3,
                               newdata=titanic_val[,2:length(titanic_train)],
                               type='response')

# computing Brier score and adding to my dataframe
performance$BrierScore[3] <-
  (1/nrow(titanic_val)) * sum((model.titanic.pred3 - outcomes_val)^2)

# providing predictions converted to outcomes using threshold of 0.5
CM3 <- confusionMatrix(data = relevel(as.factor(ifelse(model.titanic.pred3 >0.5, 1, 0)),
                                      '1'),
                       reference= titanic_val$Survived, mode='everything')

# confusion matrix
CM3$table
```

```
##           Reference
```

```
## Prediction  1  0
##          1 37 11
##          0 12 74
```

```
# storing values
performance$accuracy[3] <- CM3$overall[[1]]
performance$F1[3] <- CM3$byClass[[7]]
performance$specificity[3] <- CM3$byClass[[2]]

# viewing metrics relating to confusion matrix
CM3
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  0
##          1 37 11
##          0 12 74
##
##                Accuracy : 0.8284
##                  95% CI : (0.7537, 0.888)
##     No Information Rate : 0.6343
##     P-Value [Acc > NIR] : 6.819e-07
##
##                   Kappa : 0.6284
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.7551
##             Specificity : 0.8706
##          Pos Pred Value : 0.7708
##          Neg Pred Value : 0.8605
##               Precision : 0.7708
##                  Recall : 0.7551
##                      F1 : 0.7629
##              Prevalence : 0.3657
##          Detection Rate : 0.2761
##    Detection Prevalence : 0.3582
##       Balanced Accuracy : 0.8128
##
##        'Positive' Class : 1
##
```

```
# performance on models
performance
```

```
##                             modelName BICscore BrierScore  accuracy        F1
## 1      model.titanic1, no interacitons 742.3893  0.1244595 0.8432836 0.7835052
## 2 model.titanic2.step, no interactions 714.3441  0.1328708 0.8059701 0.7450980
## 3      model.titanic3, no interactions 729.8140  0.1277599 0.8283582 0.7628866
##   specificity
## 1   0.8823529
## 2   0.8235294
## 3   0.8705882
```

This model, model 3, seems to have found a position between the two previous models. It's BIC score improves on the first model while the Brier Score is slightly worse than the first. It's accuracy, F1, and specificity are between the two previous models, with specificity (True Negative Rate) being very close to the first.

Interactions Now I will build a final model with interaction effects. I postulate that interactions of PClass and Title will have a strong effect on probability of survival, but this remains to be seen. I am going to remove columns that I am now confident I won't be using. I will not be using the SibSp variable (siblings and spouses onboard) and the 5OrMoreSibSp variable (passenger has 5 or more siblings and spouses onboard). Then I will use the step function to build this model that includes interactions.

```
titanic_train <- titanic_train[,
                               -which(names(titanic_train) %in% c("SibSp", "FiveOrMoreSibSp"))]
titanic_val <- titanic_val[,
                           -which(names(titanic_val) %in% c("SibSp", "FiveOrMoreSibSp"))]
```

```
# scope for interactions
null.int <- glm(Survived~1, data=titanic_train, family='binomial')
full.int <- glm(Survived~.^2, data=titanic_train, family='binomial')
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
model.titanic4.step <- step(null.int, scope = formula(full.int), direction='forward',
                            k = log(nrow(titanic_train)), trace=F)
summary(model.titanic4.step)
```

```
##
## Call:
## glm(formula = Survived ~ Title + Pclass + NoSibSp + Pclass:NoSibSp,
##     family = "binomial", data = titanic_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.5406  -0.5680  -0.4453   0.5927   2.7558
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)        3.12829    0.57440   5.446 5.15e-08 ***
## TitleMiss         -0.12712    0.46744  -0.272 0.785665
## TitleMr           -3.33424    0.48824  -6.829 8.54e-12 ***
## TitleMrs           0.05859    0.49355   0.119 0.905505
## TitleOther        -2.91791    0.68392  -4.266 1.99e-05 ***
## Pclass2           -1.46535    0.52151  -2.810 0.004957 **
## Pclass3           -3.56848    0.45727  -7.804 6.00e-15 ***
## NoSibSp1          -0.33571    0.40001  -0.839 0.401322
## Pclass2:NoSibSp1   0.26440    0.61743   0.428 0.668482
## Pclass3:NoSibSp1   1.84890    0.51629   3.581 0.000342 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
```

```
##     Null deviance: 1010.46  on 756  degrees of freedom
## Residual deviance:  649.28  on 747  degrees of freedom
## AIC: 669.28
##
## Number of Fisher Scoring iterations: 5
```

This step selected model that includes interactions is the exact same as the step selected model that doesn't include interactions. This doesn't seem right to me. I am going to make a change to my training set. I will drop the Title variable and replace it with the Sex variable. I think that this could potentially solve some issues, as the titles are gendered besides the title Other (which does not have many values). I believe using Sex, which only has 2 categories, could yield interactions.

```
# removing Title variable
titanic_train <- titanic_train[, -which(names(titanic_train) %in% c("Title"))]
titanic_val <- titanic_val[, -which(names(titanic_val) %in% c("Title"))]

# adding back the Sex variable
titanic_train$Sex <- as.factor(data_titanic$Sex[tr])
titanic_val$Sex <- as.factor(data_titanic$Sex[-tr])
```

```
# double checking VIF scores
vif(glm(Survived~., data=titanic_train, family='binomial'))
```

```
##              GVIF Df GVIF^(1/(2*Df))
## Pclass   2.498960  2        1.257303
## Age      1.297471  1        1.139066
## Parch    1.464216  1        1.210048
## Embarked 1.285131  3        1.042696
## NoSibSp  1.331764  1        1.154021
## sqrtFare 2.176363  1        1.475250
## Sex      1.203161  1        1.096887
```

With this new variable added, I will try to get a new logistic regression model for the training data.

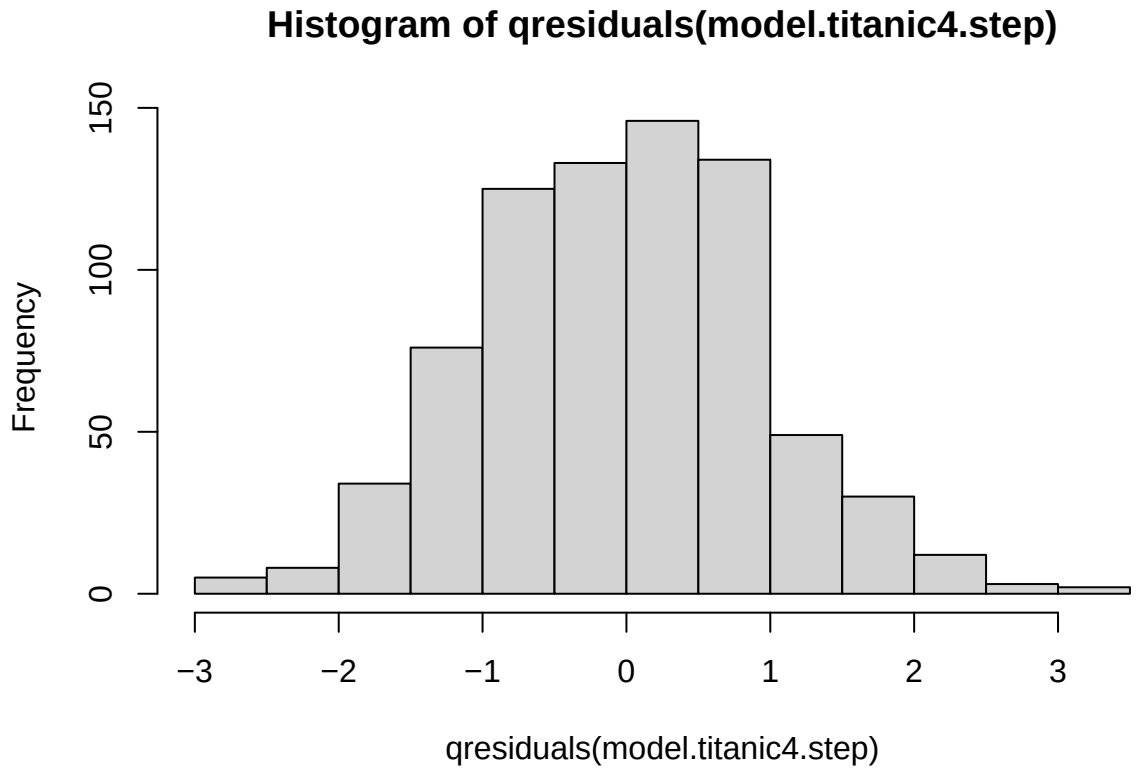
```
null.int <- glm(Survived~1, data=titanic_train, family='binomial')
full.int <- glm(Survived~.^2, data=titanic_train, family='binomial')
model.titanic4.step <- step(null.int, scope = formula(full.int), direction='forward',
                            k = log(nrow(titanic_train)), trace=F)
summary(model.titanic4.step)
```

```
##
## Call:
## glm(formula = Survived ~ Sex + Pclass + Age + Sex:Pclass, family = "binomial",
##     data = titanic_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.0601  -0.6061  -0.4950   0.4244   2.3306
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)     5.343320   1.047945   5.099 3.42e-07 ***
```

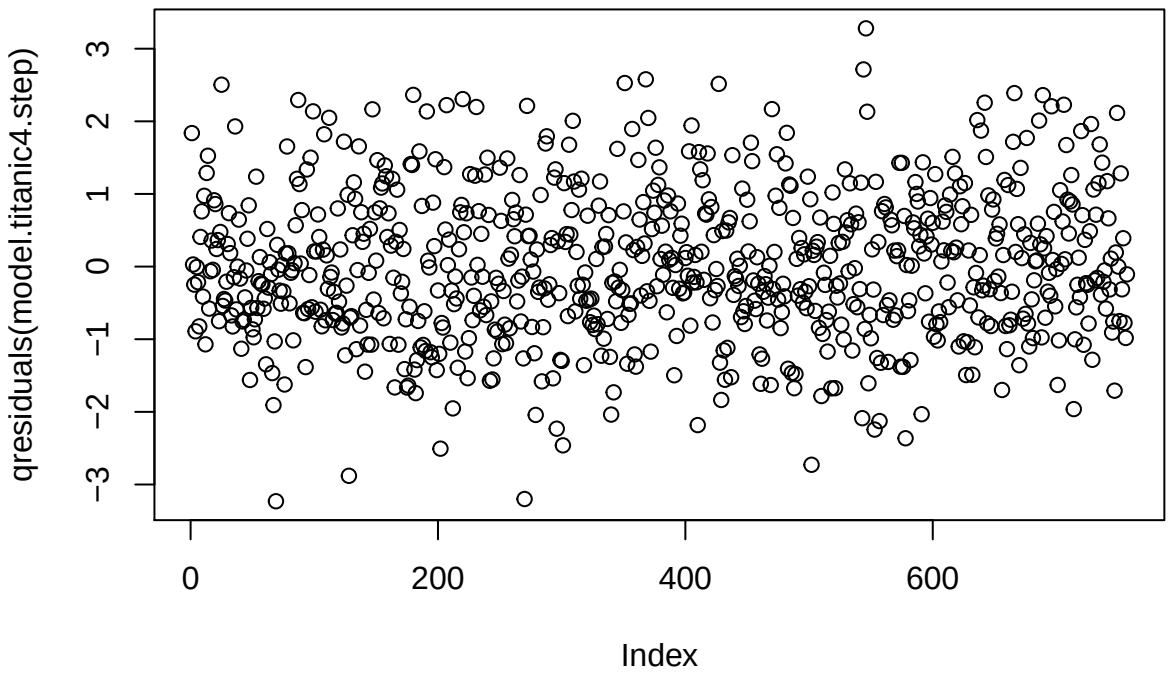
```
## Sexmale         -4.819721   1.025875  -4.698 2.63e-06 ***
## Pclass2         -2.257196   1.095845  -2.060 0.039420 *
## Pclass3         -4.670335   1.029585  -4.536 5.73e-06 ***
## Age             -0.026819   0.007479  -3.586 0.000336 ***
## Sexmale:Pclass2  0.749059   1.153957   0.649 0.516260
## Sexmale:Pclass3  3.020784   1.056222   2.860 0.004237 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1010.46  on 756  degrees of freedom
## Residual deviance:  660.66  on 750  degrees of freedom
## AIC: 674.66
##
## Number of Fisher Scoring iterations: 6
```

Checking residuals

```
hist(qresiduals(model.titanic4.step))
```




```
plot(qresiduals(model.titanic4.step))
```



Reisudal assumptions are met. The step selected model has included 2 interactions, 1 of which is significant. 3rd class male passengers have a p-value of 0.004. Let's see how this model performs compared to the rest.

```
# adding new row to the performance dataframe
performance[4,] <- NA
performance$modelName[4] <- 'model.titanic4.step, interactions, replace Title with Sex'
performance$BICscore[4] <-  extractAIC(model.titanic4.step, k=log(nrow(titanic_train)))[2]

# prediction
model.titanic.pred4 <- predict(model.titanic4.step,
                               newdata=titanic_val[,2:length(titanic_val)],
                               type='response')

# computing Brier score and adding to my dataframe
performance$BrierScore[4] <-
  (1/nrow(titanic_val)) * sum((model.titanic.pred4 - outcomes_val)^2)

# providing predictions converted to outcomes using threshold of 0.5
CM4 <- confusionMatrix(data = relevel(as.factor(ifelse(model.titanic.pred4 >0.5, 1, 0)),
                                      '1'),
                       reference= titanic_val$Survived, mode='everything')

# confusion matrix
CM4$table
```

```
##           Reference
```

```
## Prediction  1  0
##          1 32  9
##          0 17 76
```

```
# storing values
performance$accuracy[4] <- CM4$overall[[1]]
performance$F1[4] <- CM4$byClass[[7]]
performance$specificity[4] <- CM4$byClass[[2]]

# viewing metrics relating to confusion matrix
CM4
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  0
##          1 32  9
##          0 17 76
##
##                Accuracy : 0.806
##                  95% CI : (0.7288, 0.8692)
##     No Information Rate : 0.6343
##     P-Value [Acc > NIR] : 1.216e-05
##
##                   Kappa : 0.5668
##
##  Mcnemar's Test P-Value : 0.1698
##
##             Sensitivity : 0.6531
##             Specificity : 0.8941
##          Pos Pred Value : 0.7805
##          Neg Pred Value : 0.8172
##               Precision : 0.7805
##                  Recall : 0.6531
##                      F1 : 0.7111
##              Prevalence : 0.3657
##          Detection Rate : 0.2388
##    Detection Prevalence : 0.3060
##       Balanced Accuracy : 0.7736
##
##        'Positive' Class : 1
##
```

```
# performance on models
performance
```

```
##                                                modelName BICscore BrierScore
## 1                          model.titanic1, no interacitons 742.3893  0.1244595
## 2                       model.titanic2.step, no interactions 714.3441  0.1328708
## 3                          model.titanic3, no interactions 729.8140  0.1277599
## 4 model.titanic4.step, interactions, replace Title with Sex 707.0690  0.1340191
##    accuracy        F1 specificity
## 1 0.8432836 0.7835052   0.8823529
```

```
## 2 0.8059701 0.7450980   0.8235294
## 3 0.8283582 0.7628866   0.8705882
## 4 0.8059701 0.7111111   0.8941176
```

To my surprise, the interaction model does not perform better than all previous models. While it has a lower BIC score, other performance metrics show it is not aas good as previous models. It's Brier score is the worst of all the models, and it has the same accuracy as model2 (the other step model) so they share last place in that metric. It has the worse F1 score. However, it does have the best specificity value, which means it classifies passengers that do not survive better than any other model (specificity is the True Negative Rate).

Which model is my best model? Since models with lower BIC scores are clearly not the best performing, I think that they will not be useful for evaluation. I believe that model1 or model3 are the two best models, based on Having the 2 highest accuracies and F1 scores, as well as the two lowest Brier Scores. They are rank 2 and 3 for specificity, but are not far behind model 4 in that metric. Is the slight increase in complexity present in model 1 worth the increased performance over model 3? I would argue that the increased complexity is not worth the increased performance. The performance is for a small proportion of the total dataset, and I think that the chance of overfitting with more complexity present in model 1 might result in worst performance on the test set. Model 3 (titanic.model3) is my selected model, and here is its summary before proceeding to testing.

```
summary(model.titanic3)
```

```
##
## Call:
## glm(formula = Survived ~ . - SibSp - FiveOrMoreSibSp - Embarked,
##     family = "binomial", data = titanic_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3553  -0.6465  -0.3780   0.5985   2.6435
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  2.334821   0.674396   3.462 0.000536 ***
## Pclass2     -1.135957   0.348951  -3.255 0.001133 **
## Pclass3     -2.161785   0.348669  -6.200 5.64e-10 ***
## Age         -0.016647   0.008568  -1.943 0.052038 .
## Parch1      -0.660177   0.290640  -2.271 0.023119 *
## TitleMiss    0.124850   0.464321   0.269 0.788016
## TitleMr     -2.899829   0.519564  -5.581 2.39e-08 ***
## TitleMrs     0.654054   0.521538   1.254 0.209810
## TitleOther  -2.543876   0.732997  -3.471 0.000519 ***
## NoSibSp1     0.591408   0.248959   2.376 0.017524 *
## sqrtFare     0.064726   0.049457   1.309 0.190618
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1010.46  on 756  degrees of freedom
## Residual deviance:  656.89  on 746  degrees of freedom
## AIC: 678.89
##
```

```
## Number of Fisher Scoring iterations: 5
```

My most relevant predictors from my model are Passenger class, Age, having at least one parent or child onboard, the titles of Mr, Other, and Master, and having no siblings or spouses onboard.

Testing on Test Dataset

I will make a copy of the test dataset to construct the appropriate columns for prediction on the dataset.

```
# copy of test dataset
titanic_test_pred <- titanic_test
rownames(titanic_test_pred) <- titanic_test_pred$PassengerId
titanic_test_pred <-
  titanic_test_pred[,
                    -which(names(titanic_test_pred) %in% c("Name",
                                                           "PassengerId",
                                                           "Sex",
                                                           "Ticket",
                                                           "Cabin",
                                                           "AgeEst"))]
head(titanic_test_pred)
```

```
##     Pclass  Age SibSp Parch    Fare Embarked Title
## 892      3 34.5     0     0  7.8292        Q    Mr
## 893      3 47.0     1     0  7.0000        S   Mrs
## 894      2 62.0     0     0  9.6875        Q    Mr
## 895      3 27.0     0     0  8.6625        S    Mr
## 896      3 22.0     1     1 12.2875        S   Mrs
## 897      3 14.0     0     0  9.2250        S    Mr
```

```
titanic_test_pred$Pclass <- as.factor(titanic_test_pred$Pclass)
titanic_test_pred$Parch <- as.factor(ifelse(titanic_test_pred$Parch > 0 , 1, 0))

titanic_test_pred$NoSibSp <- ifelse(titanic_test_pred$SibSp == 0, 1, 0)
titanic_test_pred$NoSibSp <- as.factor(titanic_test_pred$NoSibSp)

titanic_test_pred$FiveOrMoreSibSp <- ifelse(titanic_test_pred$SibSp>4,1,0)
titanic_test_pred$FiveOrMoreSibSp <- as.factor(titanic_test_pred$FiveOrMoreSibSp)

titanic_test_pred$Title <- as.factor(titanic_test_pred$Title)
titanic_test_pred$sqrtFare <- sqrt(titanic_test_pred$Fare)
titanic_test_pred <- titanic_test_pred[, -which(names(titanic_test_pred)
                                                %in% c("Fare"))]

titanic_test_pred$Embarked <- as.factor(titanic_test_pred$Embarked)
names(titanic_test_pred)
```

```
## [1] "Pclass"          "Age"             "SibSp"           "Parch"
## [5] "Embarked"        "Title"           "NoSibSp"         "FiveOrMoreSibSp"
## [9] "sqrtFare"
```

Now prediciton for the test data can be done.

```r
# predictions
final_predictions <- predict(model.titanic3, newdata=titanic_test_pred, type='response')

# outcomes using threshold of 500

sum(as.numeric(data_titanic$Survived))
```

```
## [1] 342
```

Below is the outcomes from the predicted probabilties. Given that the train and test datasets have 342 survivors, and there were about 500 surviving passenger on the ship, I am hoping to see a value near 158 to show good performance from my model.

```r
# number of survivors in training and test dataset
sum(data_titanic$Survived)
```

```
## [1] 342
```

```r
# expected survivors in this dataset
500-342
```

```
## [1] 158
```

```r
outcomes_test <- ifelse(final_predictions>0.5, 1, 0)
sum(outcomes_test)
```

```
## [1] 154
```

```r
print('Number of Survivors in Train/Validate Dataset:  342')
```

```
## [1] "Number of Survivors in Train/Validate Dataset:  342"
```

```r
print('Expected Number of Survivors in Test Dataset: 158')
```

```
## [1] "Expected Number of Survivors in Test Dataset: 158"
```

```r
print('Number of Predicted Survivors in Test Dataset: 154')
```

```
## [1] "Number of Predicted Survivors in Test Dataset: 154"
```

Wow! My model was able to classify 154 passengers as survivors, compared to a groundtruth total of about 158.. While there could false negatives and false positives, I do not have the groundtruth survival statistic for each individual passenger and cannot check.

Let's add these outcomes for the passengers to the prediction dataframe, and see the number of survivors, grouped by the strong predictors.

```
titanic_test_pred$Survived <- outcomes_test
```

```
titanic_test_pred$Age_Range <- ifelse(titanic_test_pred$Age <= 18,
                                      '18 or less', 'Over 18')
titanic_test_pred$ParentOrChildOnboard <- ifelse(titanic_test_pred$Parch == 1,
                                                 'yes', 'no')
titanic_test_pred$SiblingsOrSpouseOnboard <- ifelse(titanic_test_pred$NoSibSp == 1,
                                                    'no', 'yes')
# creating a dataframe that groups by the list of variables in the 'by' argument,
# and shows the number of passengers that survived in each group.
group_survivals <-
  aggregate(titanic_test_pred$Survived,
            by=list(Age_Range = titanic_test_pred$Age_Range,
                    Class=titanic_test_pred$Pclass,
                    Title= titanic_test_pred$Title,
                    ParentOrChildOnboard =titanic_test_pred$ParentOrChildOnboard,
                    SiblingsOrSpouse = titanic_test_pred$SiblingsOrSpouseOnboard),
                      FUN=sum)

# now taking the number of deaths for each group. I am taking the sum of the absolute
# value of the Survived column minus 1, which will make all survivors be assigned 0,
# and those that didnt negative.
group_deaths <-
  aggregate(abs(titanic_test_pred$Survived-1),
            by=list(Age_Range = titanic_test_pred$Age_Range,
                    Class=titanic_test_pred$Pclass,
                    Title= titanic_test_pred$Title,
                    ParentOrChildOnboard = titanic_test_pred$ParentOrChildOnboard,
                    SiblingsOrSpouse = titanic_test_pred$SiblingsOrSpouseOnboard),
                  FUN=sum)

group_survivals$Deaths <- group_deaths$x

names(group_survivals) <- c(names(group_survivals)[1:(length(group_survivals)-2)],
                            'Survivors', 'Deaths')

summary(model.titanic3)
```

```
##
## Call:
## glm(formula = Survived ~ . - SibSp - FiveOrMoreSibSp - Embarked,
##     family = "binomial", data = titanic_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3553  -0.6465  -0.3780   0.5985   2.6435
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  2.334821   0.674396   3.462 0.000536 ***
## Pclass2     -1.135957   0.348951  -3.255 0.001133 **
## Pclass3     -2.161785   0.348669  -6.200 5.64e-10 ***
## Age         -0.016647   0.008568  -1.943 0.052038 .
```

```
## Parch1      -0.660177   0.290640  -2.271 0.023119 *
## TitleMiss    0.124850   0.464321   0.269 0.788016
## TitleMr     -2.899829   0.519564  -5.581 2.39e-08 ***
## TitleMrs     0.654054   0.521538   1.254 0.209810
## TitleOther  -2.543876   0.732997  -3.471 0.000519 ***
## NoSibSp1     0.591408   0.248959   2.376 0.017524 *
## sqrtFare     0.064726   0.049457   1.309 0.190618
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1010.46  on 756  degrees of freedom
## Residual deviance:  656.89  on 746  degrees of freedom
## AIC: 678.89
##
## Number of Fisher Scoring iterations: 5
```

```
names(titanic_test_pred)
```

```
##  [1] "Pclass"                 "Age"
##  [3] "SibSp"                  "Parch"
##  [5] "Embarked"               "Title"
##  [7] "NoSibSp"                "FiveOrMoreSibSp"
##  [9] "sqrtFare"               "Survived"
## [11] "Age_Range"              "ParentOrChildOnboard"
## [13] "SiblingsOrSpouseOnboard"
```

```
group_survivals
```

```
##     Age_Range Class  Title ParentOrChildOnboard SiblingsOrSpouse Survivors
## 1  18 or less     3 Master                   no               no         1
## 2     Over 18     1   Miss                   no               no        11
## 3  18 or less     2   Miss                   no               no         1
## 4     Over 18     2   Miss                   no               no         4
## 5  18 or less     3   Miss                   no               no        10
## 6     Over 18     3   Miss                   no               no        24
## 7  18 or less     1     Mr                   no               no         1
## 8     Over 18     1     Mr                   no               no         4
## 9  18 or less     2     Mr                   no               no         0
## 10    Over 18     2     Mr                   no               no         0
## 11 18 or less     3     Mr                   no               no         0
## 12    Over 18     3     Mr                   no               no         0
## 13    Over 18     1    Mrs                   no               no         6
## 14    Over 18     2    Mrs                   no               no         5
## 15 18 or less     3    Mrs                   no               no         1
## 16    Over 18     3    Mrs                   no               no         4
## 17    Over 18     1  Other                   no               no         1
## 18    Over 18     2  Other                   no               no         0
## 19    Over 18     3  Other                   no               no         0
## 20 18 or less     1 Master                  yes               no         1
## 21 18 or less     2 Master                  yes               no         1
## 22 18 or less     3 Master                  yes               no         4
```

```
## 23    Over 18     1   Miss                yes               no         2
## 24 18 or less     2   Miss                yes               no         1
## 25    Over 18     2   Miss                yes               no         1
## 26 18 or less     3   Miss                yes               no         1
## 27    Over 18     1     Mr                yes               no         0
## 28    Over 18     2     Mr                yes               no         0
## 29    Over 18     3     Mr                yes               no         0
## 30    Over 18     1    Mrs                yes               no         6
## 31    Over 18     2    Mrs                yes               no         4
## 32    Over 18     3    Mrs                yes               no         5
## 33    Over 18     2   Miss                 no              yes         2
## 34 18 or less     3   Miss                 no              yes         1
## 35    Over 18     3   Miss                 no              yes         3
## 36    Over 18     1     Mr                 no              yes         0
## 37    Over 18     2     Mr                 no              yes         0
## 38 18 or less     3     Mr                 no              yes         0
## 39    Over 18     3     Mr                 no              yes         0
## 40 18 or less     1    Mrs                 no              yes         2
## 41    Over 18     1    Mrs                 no              yes        13
## 42    Over 18     2    Mrs                 no              yes         4
## 43    Over 18     3    Mrs                 no              yes         6
## 44    Over 18     1  Other                 no              yes         0
## 45 18 or less     1 Master                yes              yes         1
## 46 18 or less     2 Master                yes              yes         1
## 47 18 or less     3 Master                yes              yes         0
## 48    Over 18     1   Miss                yes              yes         1
## 49 18 or less     2   Miss                yes              yes         5
## 50    Over 18     2   Miss                yes              yes         2
## 51 18 or less     3   Miss                yes              yes         0
## 52    Over 18     3   Miss                yes              yes         0
## 53    Over 18     1     Mr                yes              yes         0
## 54    Over 18     2     Mr                yes              yes         0
## 55 18 or less     3     Mr                yes              yes         0
## 56    Over 18     3     Mr                yes              yes         0
## 57    Over 18     1    Mrs                yes              yes         8
## 58    Over 18     2    Mrs                yes              yes         1
## 59 18 or less     3    Mrs                yes              yes         2
## 60    Over 18     3    Mrs                yes              yes         3
## 61    Over 18     1  Other                yes              yes         0
## 62    Over 18     2  Other                yes              yes         0
##    Deaths
## 1       0
## 2       0
## 3       0
## 4       0
## 5       0
## 6       0
## 7       0
## 8      27
## 9       4
## 10     39
## 11      6
## 12    100
## 13      0
```

```
## 14      0
## 15      0
## 16      0
## 17      1
## 18      1
## 19      1
## 20      0
## 21      0
## 22      0
## 23      0
## 24      0
## 25      0
## 26      0
## 27      1
## 28      1
## 29      2
## 30      0
## 31      0
## 32      0
## 33      0
## 34      0
## 35      0
## 36     12
## 37     12
## 38      2
## 39     13
## 40      0
## 41      0
## 42      0
## 43      0
## 44      1
## 45      0
## 46      0
## 47     12
## 48      0
## 49      0
## 50      0
## 51      7
## 52      2
## 53      7
## 54      3
## 55      1
## 56      4
## 57      0
## 58      0
## 59      0
## 60      2
## 61      1
## 62      1
```

Conclusions

With my analysis concluded, I can finally answer my questions of interest. At the start of this analysis I asked, is the provided information in this titanic passenger dataset enough to accurately classify survivors? And if so, which of these fields provided are stronger predictors than the others? Using my model (a logistic regression model), I have confidence that I can correctly answer these questions. First, the provided information is without a doubt enough to accurately classify passengers. I trained my model on only about 58% of the total passenger data, and validated it on about 10% of the total data, with the remaining approximatley 32% being used for testing. Despite a suboptimal proportion of data that was used for training, there were overall strong confusion matrix performance metrics for my final model on the validation set. I was able to leverage some feature engineering in order to expand the possible information that could be used for each passenger.

Second, there were some information fields of the passengers that were more significant than others for the task of classifying the passengers as survivors. Sex, which was moreso represented by the field Title (Mr, Mrs, Miss, Master, or Other) was statistically significant. Passenger ticket class (1st, 2nd or 3rd) was also statistically significant. 3rd and 2nd class passengers had a much less likely probability of survival as their negative regression coefficients indicate. Passengers with parents or children onboard had a decreased chance of survival, and this field was also a strong indicator of survival. Finally, having no siblings or spouses onboard increased probability of survival, but this impact on survival was not as strong as the aforementioned fields.