Final Project 1361

Daniel Antantis

```
library(tidyverse)
library(glmnet)
library(gam)
library(boot)
library(leaps)
library(randomForest)
library(caret)
```

Read in Data

```
raw_data = read_csv("~/Documents/STAT1361/train.csv")
```

```
## Rows: 6552 Columns: 15

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (4): Date, Seasons, Holiday, Functioning
## dbl (11): Count, Hour, Temperature, Humidity, Wind, Visibility, Dew, Solar, ...

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Data Wrangling

```
#Separate the date variable into three separate variables because I want to see if individually they ar
out.data = raw_data %>%
  separate(Date, sep="/", into = c("Day", "Month", "Year"))
```

```
summary(out.data$Count)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     0.0   189.0   492.0   702.9  1062.0  3556.0
```

```
#Create percentiles to extract outliers in the data with extreme count values
lower_bound = quantile(out.data$Count, 0.01)
upper_bound = quantile(out.data$Count, 0.99)

outliers.ind = which(out.data$Count < lower_bound | out.data$Count > upper_bound)

out.data[outliers.ind, ]
```

```
## # A tibble: 66 x 17
##    Count Day   Month Year   Hour Temperature Humidity  Wind Visibility   Dew
##    <dbl> <chr> <chr> <chr> <dbl>       <dbl>    <dbl> <dbl>      <dbl> <dbl>
##  1  2692 16    4     2018     18        17         28   3.1       2000  -1.6
##  2  2807 25    4     2018     18        21.2       32   3.8       1927   3.8
##  3  2574 26    4     2018     18        17.4       45   3.1       1092   5.3
##  4  2661 4     5     2018     18        17.1       35   3.4       1961   1.4
##  5  3130 9     5     2018     18        20.6       41   2.3       2000   6.8
##  6  2701 11    5     2018     18        17.9       37   3.1       1819   2.9
##  7  2906 14    5     2018     18        23.6       48   3.1        666  11.9
##  8  3069 21    5     2018     18        21.6       48   2.5       1884  10.1
##  9  3123 23    5     2018     18        21.7       40   3.5       1987   7.4
## 10  2916 25    5     2018     18        23.3       32   2.6       1772   5.6
## # ... with 56 more rows, and 7 more variables: Solar <dbl>, Rainfall <dbl>,
## #   Snowfall <dbl>, Seasons <chr>, Holiday <chr>, Functioning <chr>, ID <dbl>
```

```
#Explore the outliers and look for trends that may help explain the extreme values
range(out.data[outliers.ind, ]$Temperature)
```

```
## [1] 14.3 33.0
```

```
range(out.data[outliers.ind, ]$Humidity)
```

```
## [1] 27 77
```

```
range(out.data[outliers.ind, ]$Hour)
```

```
## [1] 17 20
```

```
mean(out.data[outliers.ind, ]$Temperature)
```

```
## [1] 24.97121
```

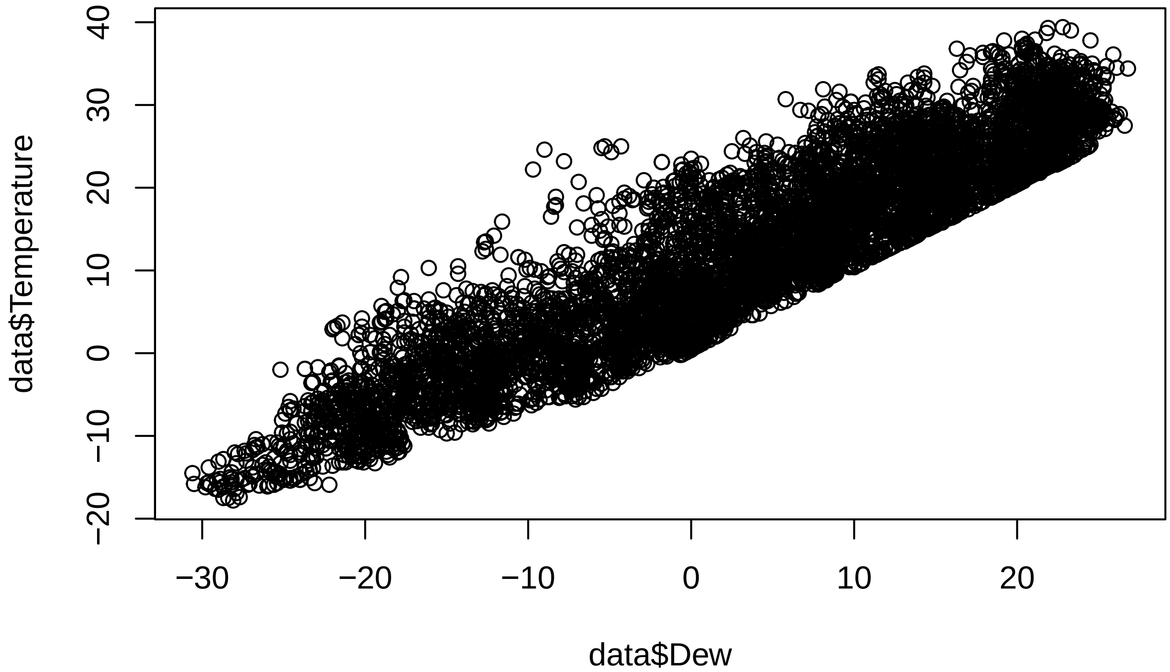
```
mean(out.data[outliers.ind, ]$Humidity)
```

```
## [1] 49.18182
```

```
#Remove the outliers from the dataset
data = out.data[-outliers.ind, ]
```

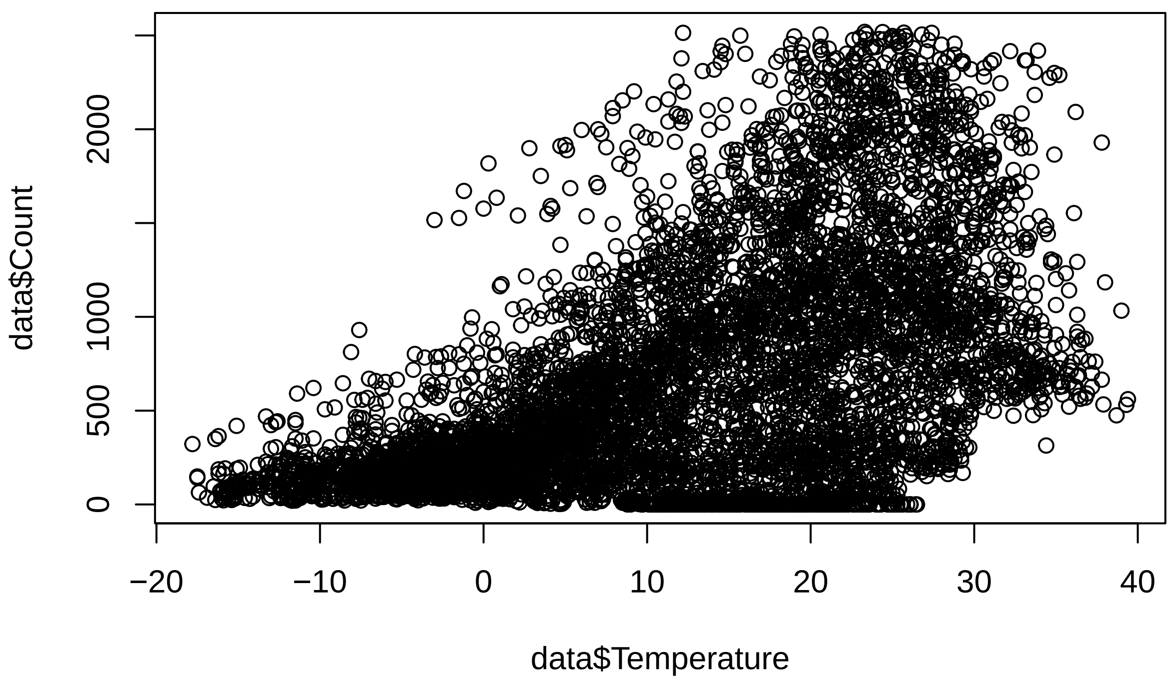
Colinearity

```
#Explore the strong colinearity between dew point temperature and temperature as they are both measurin
plot(data$Dew, data$Temperature)
```

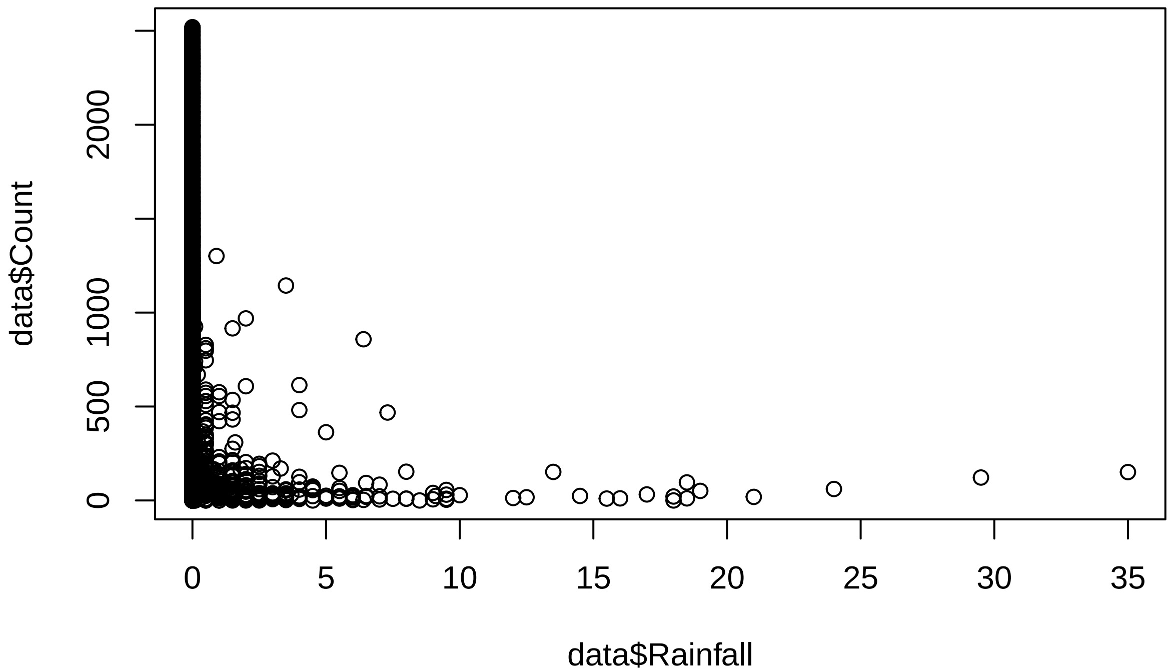


Relationship Analysis

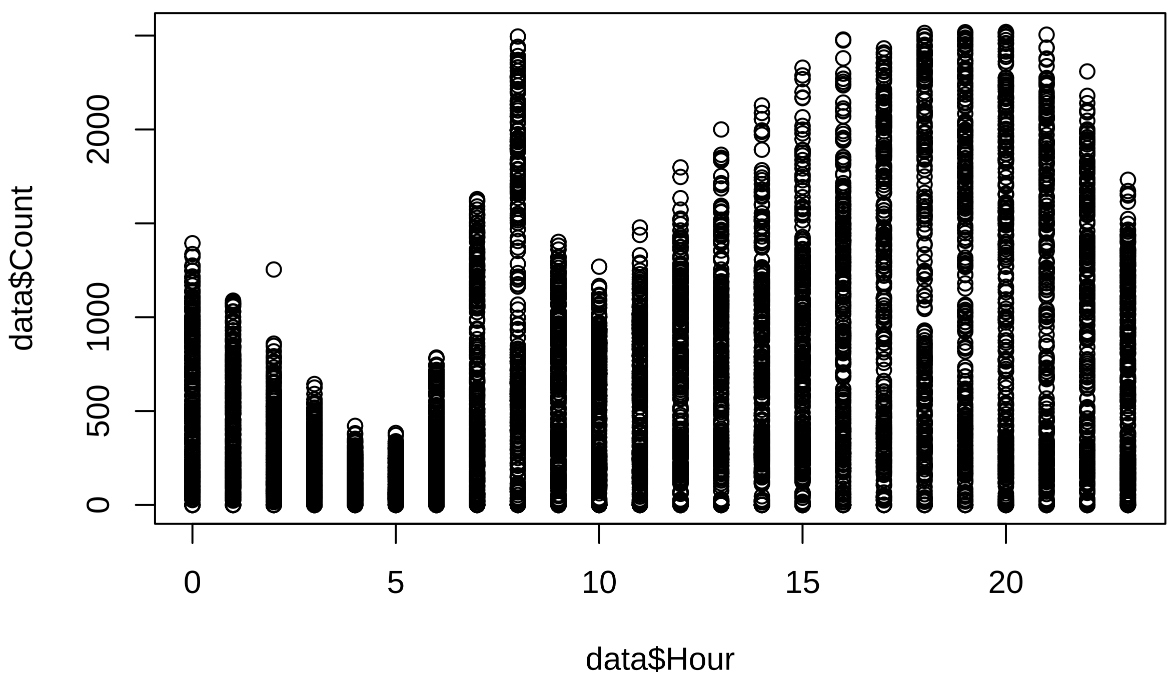
```
plot(data$Temperature, data$Count)
```



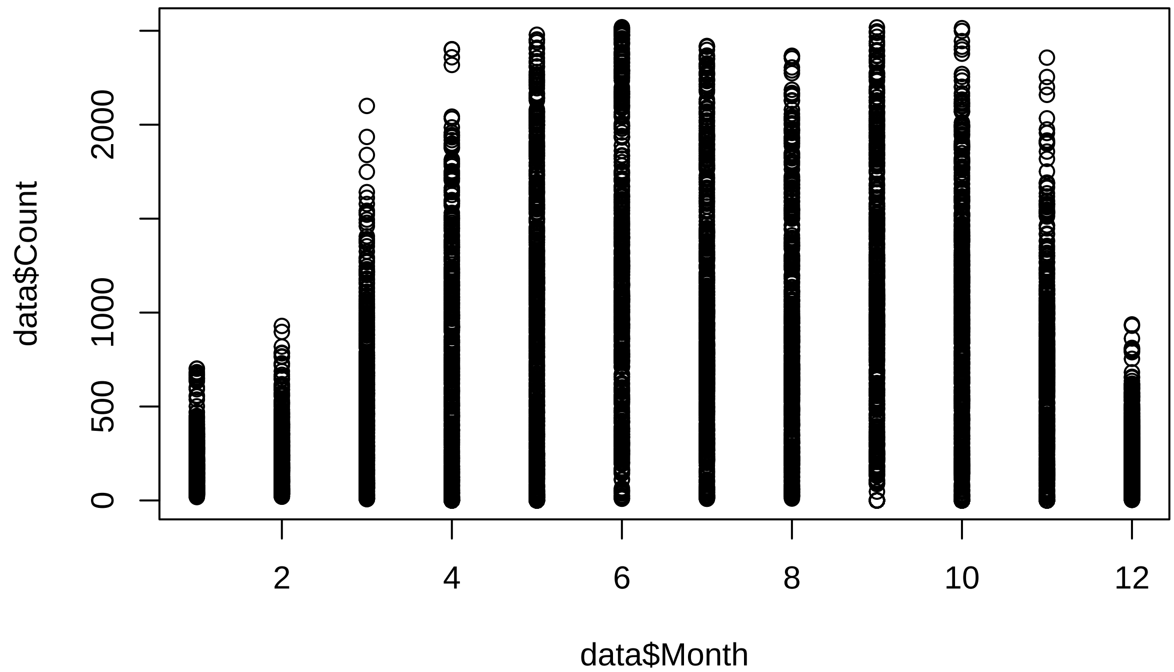

```
plot(data$Rainfall, data$Count)
```



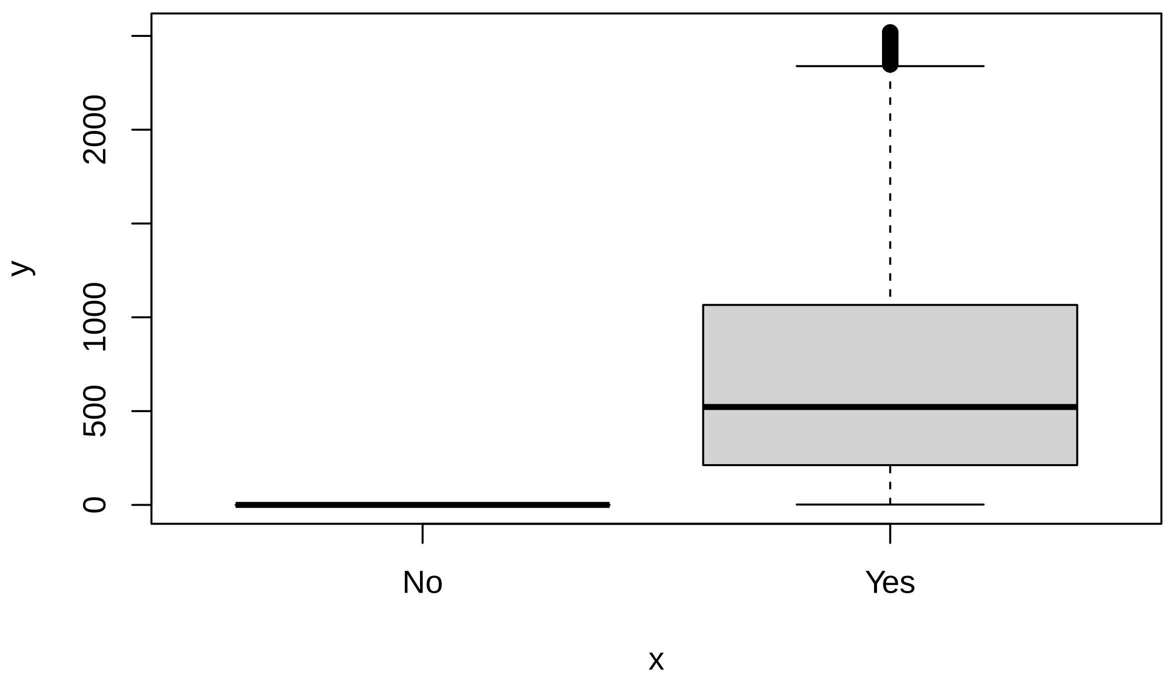

```
plot(data$Hour, data$Count)
```



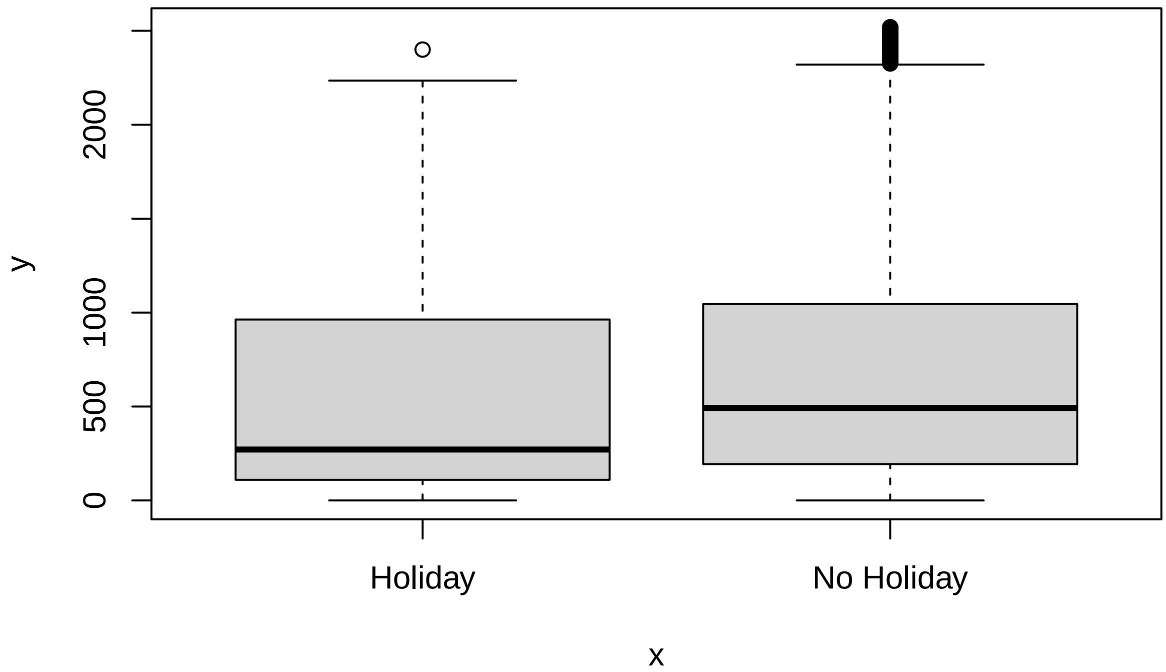

```
plot(data$Month, data$Count)
```



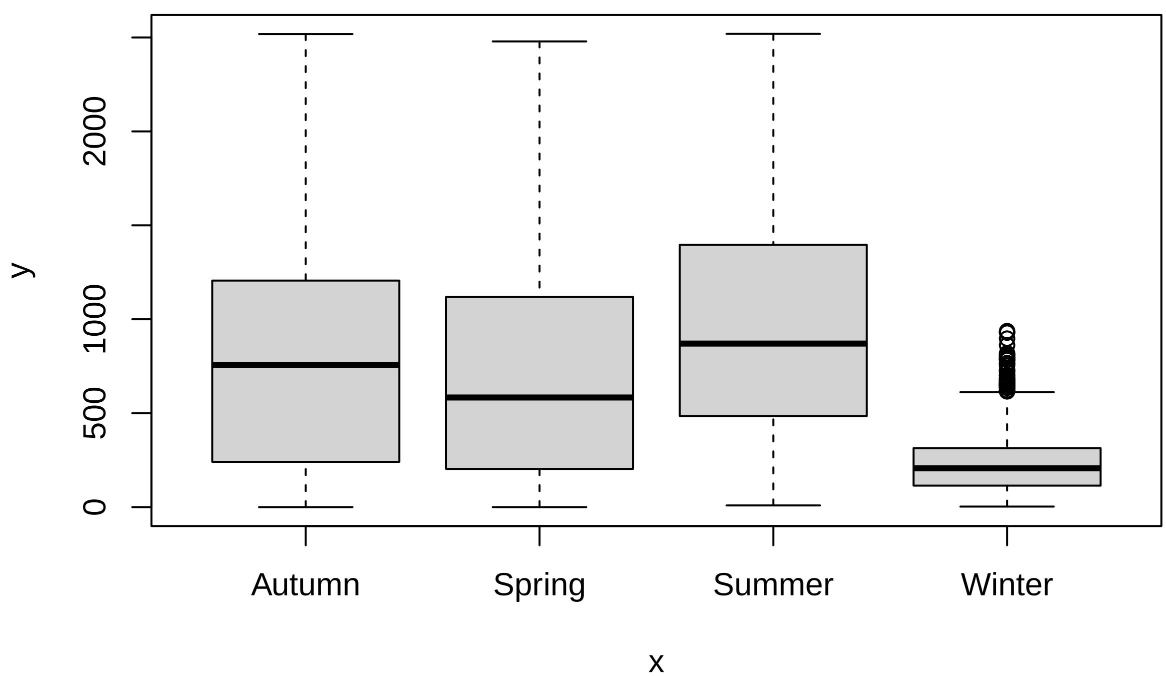

```
#Explore the functioning variable and see that it has a very strong affect on bike rental count therfor
plot(as.factor(data$Functioning), data$Count)
```




```
#Extra plots of some of the categorical variables
plot(as.factor(data$Holiday), data$Count)
```




```
plot(as.factor(data$Seasons), data$Count)
```



Split

Data

```
#Splitting the data into training and testing dataset with the training set containing 75% of the data
set.seed(21)
sampleSize = floor(0.75 * nrow(data))
split = sample(seq_len(nrow(data)), size = sampleSize)
train = data[split, ]
test = data[-split, ]
```

Linear Model

```
#Test all of the variables in the dataset and see which ones standout
lin.fit.test = lm(Count~Hour+Temperature+Humidity+Wind+Visibility+Dew+Solar+Rainfall+Snowfall+Holiday+Mo
summary(lin.fit.test)
```

```
##
## Call:
## lm(formula = Count ~ Hour + Temperature + Humidity + Wind + Visibility +
##     Dew + Solar + Rainfall + Snowfall + Holiday + Month + Year +
##     Day + Functioning, data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1264.40  -251.12   -39.11   207.74  1561.27
##
## Coefficients: (1 not defined because of singularities)
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -760.55654  118.80561  -6.402 1.68e-10 ***
## Hour                24.96820    0.90374  27.628  < 2e-16 ***
## Temperature         25.55621    4.39474   5.815 6.45e-09 ***
## Humidity            -7.49874    1.20431  -6.227 5.17e-10 ***
## Wind                12.03098    6.04632   1.990 0.046669 *
## Visibility           0.07947    0.01456   5.458 5.04e-08 ***
## Dew                  4.12105    4.58771   0.898 0.369081
## Solar              -70.07145    9.24739  -7.577 4.20e-14 ***
## Rainfall           -47.40677    4.86234  -9.750  < 2e-16 ***
## Snowfall            47.59727   12.76705   3.728 0.000195 ***
## HolidayNo Holiday   83.51357   28.59155   2.921 0.003506 **
## Month10            332.23645   39.77814   8.352  < 2e-16 ***
## Month11            257.20590   32.35118   7.950 2.30e-15 ***
## Month12             47.15079   26.76941   1.761 0.078240 .
## Month2             -31.31106   27.99987  -1.118 0.263513
## Month3             103.44028   32.95394   3.139 0.001706 **
## Month4             200.71836   38.00621   5.281 1.34e-07 ***
## Month5             285.85742   42.40298   6.741 1.75e-11 ***
## Month6             355.34173   49.57794   7.167 8.81e-13 ***
## Month7               4.18473   56.55410   0.074 0.941017
## Month8            -167.12984   59.31014  -2.818 0.004854 **
## Month9             239.27465   49.95153   4.790 1.72e-06 ***
## Year2018                  NA         NA      NA       NA
## Day10               14.82557   42.80940   0.346 0.729121
## Day11               98.76443   47.25837   2.090 0.036681 *
## Day12              -17.47042   44.86352  -0.389 0.696988
## Day13              144.16322   41.48588   3.475 0.000515 ***
## Day14               85.66177   42.42476   2.019 0.043527 *
## Day15               -5.47481   42.54965  -0.129 0.897625
## Day16               22.80189   43.24139   0.527 0.597998
## Day17               13.28926   42.81167   0.310 0.756261
## Day18              -29.98797   42.19753  -0.711 0.477331
## Day19               64.45597   42.60567   1.513 0.130384
## Day2               -61.56587   45.12660  -1.364 0.172539
## Day20               14.31648   41.01831   0.349 0.727085
## Day21              -14.56142   44.20499  -0.329 0.741863
```

```
## Day22              -8.87794   46.64303  -0.190 0.849052
## Day23             -15.70737   43.35961  -0.362 0.717175
## Day24             -92.18446   43.67347  -2.111 0.034844 *
## Day25              34.52368   40.15148   0.860 0.389922
## Day26             -83.67984   43.15031  -1.939 0.052528 .
## Day27              13.62328   42.39415   0.321 0.747961
## Day28             -75.13244   42.09606  -1.785 0.074359 .
## Day29             -32.81023   44.27366  -0.741 0.458682
## Day3              113.47540   48.89570   2.321 0.020341 *
## Day30              11.48323   50.68646   0.227 0.820780
## Day31             -31.70084   53.29614  -0.595 0.552002
## Day4               -0.95564   40.80279  -0.023 0.981315
## Day5               18.04469   42.67138   0.423 0.672405
## Day6               29.07477   40.95955   0.710 0.477837
## Day7              105.63816   45.28931   2.333 0.019714 *
## Day8               15.69984   46.30124   0.339 0.734564
## Day9               40.59825   41.30816   0.983 0.325748
## FunctioningYes    974.15560   31.99808  30.444  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 386 on 4811 degrees of freedom
## Multiple R-squared:  0.607,  Adjusted R-squared:  0.6027
## F-statistic: 142.9 on 52 and 4811 DF,  p-value: < 2.2e-16
```

```r
#The final linear model chosen to represent the data
lin.fit = lm(Count~Hour+Temperature+Humidity+Visibility+Solar+Rainfall+Snowfall+Holiday+Month+Day+Functi
summary(lin.fit)
```

```
##
## Call:
## lm(formula = Count ~ Hour + Temperature + Humidity + Visibility +
##     Solar + Rainfall + Snowfall + Holiday + Month + Day + Functioning,
##     data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1262.46  -251.26   -38.17   209.38  1553.56
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -827.20366   67.96109 -12.172  < 2e-16 ***
## Hour                25.38570    0.87557  28.993  < 2e-16 ***
## Temperature         29.18442    1.49524  19.518  < 2e-16 ***
## Humidity            -6.55041    0.49361 -13.270  < 2e-16 ***
## Visibility           0.08148    0.01451   5.617 2.05e-08 ***
## Solar              -67.75729    8.76779  -7.728 1.32e-14 ***
## Rainfall           -47.48225    4.83397  -9.823  < 2e-16 ***
## Snowfall            47.11824   12.75186   3.695 0.000222 ***
## HolidayNo Holiday   82.39427   28.59451   2.881 0.003976 **
## Month10            330.18588   39.63599   8.330  < 2e-16 ***
## Month11            253.13946   32.29785   7.838 5.61e-15 ***
## Month12             45.25413   26.76057   1.691 0.090887 .
## Month2             -31.85186   27.98843  -1.138 0.255162
```

```
## Month3              106.11039   32.93354   3.222 0.001282 **
## Month4              202.83947   37.99601   5.338 9.81e-08 ***
## Month5              287.57442   42.17512   6.819 1.03e-11 ***
## Month6              358.39184   49.33155   7.265 4.33e-13 ***
## Month7                6.57068   56.05174   0.117 0.906687
## Month8             -162.43348   58.97468  -2.754 0.005904 **
## Month9              238.15703   49.65426   4.796 1.66e-06 ***
## Day10                17.82049   42.79883   0.416 0.677152
## Day11               103.07326   47.20863   2.183 0.029058 *
## Day12               -21.20319   44.83229  -0.473 0.636274
## Day13               142.53796   41.47330   3.437 0.000593 ***
## Day14                84.01504   42.40193   1.981 0.047604 *
## Day15                -8.40786   42.53651  -0.198 0.843318
## Day16                20.81610   43.23398   0.481 0.630201
## Day17                10.38756   42.80054   0.243 0.808250
## Day18               -37.96349   42.04372  -0.903 0.366596
## Day19                61.41300   42.58258   1.442 0.149308
## Day2                -65.37097   45.10483  -1.449 0.147316
## Day20                13.49517   40.99847   0.329 0.742047
## Day21               -19.28412   44.13812  -0.437 0.662201
## Day22                -8.33717   46.64414  -0.179 0.858149
## Day23               -14.20174   43.34239  -0.328 0.743180
## Day24               -90.89819   43.65635  -2.082 0.037383 *
## Day25                32.95142   40.15176   0.821 0.411874
## Day26               -82.95918   43.15008  -1.923 0.054593 .
## Day27                11.85714   42.39659   0.280 0.779741
## Day28               -77.93300   42.07997  -1.852 0.064084 .
## Day29               -34.42829   44.26885  -0.778 0.436779
## Day3                111.05493   48.87439   2.272 0.023115 *
## Day30                 8.51746   50.68139   0.168 0.866544
## Day31               -32.82581   53.30368  -0.616 0.538038
## Day4                  0.37183   40.80954   0.009 0.992731
## Day5                 21.23445   42.65761   0.498 0.618656
## Day6                 30.02392   40.96854   0.733 0.463684
## Day7                107.97817   45.28481   2.384 0.017144 *
## Day8                 14.84240   46.30875   0.321 0.748596
## Day9                 36.83117   41.28071   0.892 0.372324
## FunctioningYes      969.54352   31.93579  30.359  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 386.1 on 4813 degrees of freedom
## Multiple R-squared:  0.6066, Adjusted R-squared:  0.6025
## F-statistic: 148.4 on 50 and 4813 DF,  p-value: < 2.2e-16
```

```
lin.pred = predict(lin.fit, test, type = 'response')
mean((lin.pred - test$Count)^2)
```

```
## [1] 155040.3
```

Ridge

```r
#In order to do ridge regression and lasso we must first convert the data into matrices and vectors
train.x = model.matrix(Count~Hour+Temperature+Humidity+Visibility+Solar+Rainfall+Snowfall+Holiday+Month+
train.count = train$Count
test.x = model.matrix(Count~Hour+Temperature+Humidity+Visibility+Solar+Rainfall+Snowfall+Holiday+Month+D
test.count = test$Count

ridge.fit = cv.glmnet(train.x, train.count, alpha = 0)
ridge.lambda = ridge.fit$lambda.min

ridge.pred = predict(ridge.fit, s = ridge.lambda, newx = test.x)
mean((ridge.pred - test.count)^2)
```

```
## [1] 156542.1
```

Coefficients

```r
#Examine the coefficient values of ridge regession and see where the model is weighted most heavily
ridge.coef = predict(ridge.fit, type = "coefficients", s = ridge.lambda)
ridge.coef
```

```
## 52 x 1 sparse Matrix of class "dgCMatrix"
##                              s1
## (Intercept)       -756.41984357
## (Intercept)          .
## Hour                25.20441115
## Temperature         24.53450266
## Humidity            -5.29664146
## Visibility           0.08594388
## Solar              -35.78087928
## Rainfall           -46.65097937
## Snowfall            25.88587586
## HolidayNo Holiday   73.89189041
## Month10            291.57099432
## Month11            214.94656223
## Month12            -14.79950985
## Month2             -86.69024081
## Month3              65.21893162
## Month4             169.54792121
## Month5             270.30546867
## Month6             360.83456055
## Month7              38.73290977
## Month8            -113.91267232
## Month9             234.81899205
## Day10                3.29180952
## Day11               68.68700150
## Day12              -38.11303694
## Day13              120.04840758
## Day14               74.66014605
## Day15               -9.36159380
## Day16               13.70178452
## Day17                2.88485268
## Day18              -36.66302242
```

```
## Day19                 51.59497727
## Day2                 -82.69391669
## Day20                 11.62590134
## Day21                -14.45294922
## Day22                -18.89173925
## Day23                -23.48640811
## Day24                -97.40460716
## Day25                 24.72808813
## Day26                -81.62528099
## Day27                  4.86088264
## Day28                -86.26413647
## Day29                -40.06885031
## Day3                  92.95100726
## Day30                 -0.85755125
## Day31                -26.51280702
## Day4                 -17.40438166
## Day5                   7.06520903
## Day6                   7.73930999
## Day7                  96.87295918
## Day8                  -6.48585885
## Day9                  11.45451269
## FunctioningYes       895.37553836
```

Lasso

```r
lasso.fit = cv.glmnet(train.x, train.count, alpha = 1)
lasso.lambda = lasso.fit$lambda.min

lasso.pred = predict(lasso.fit, s = lasso.lambda, newx = test.x)
mean((lasso.pred - test.count)^2)
```

```
## [1] 155180.7
```

Coefficients

```r
lasso.coef = predict(lasso.fit, type = "coefficients", s = lasso.lambda)
lasso.coef
```

```
## 52 x 1 sparse Matrix of class "dgCMatrix"
##                                 s1
## (Intercept)        -798.7363757
## (Intercept)            .
## Hour                 25.1942493
## Temperature          29.8215795
## Humidity             -6.1829193
## Visibility            0.0848889
## Solar               -62.5334573
## Rainfall            -47.1312742
## Snowfall             39.6038593
## HolidayNo Holiday    76.7807117
## Month10             284.3979813
## Month11             218.8138534
## Month12              17.9091067
```

```
## Month2              -51.6216511
## Month3               67.0238220
## Month4              157.7576781
## Month5              239.7138166
## Month6              306.5343998
## Month7              -43.0377142
## Month8             -211.5701176
## Month9              186.2000560
## Day10                 2.5363093
## Day11                87.8969593
## Day12               -23.2194963
## Day13               125.2025118
## Day14                65.7520486
## Day15               -12.8769955
## Day16                 4.8249558
## Day17                 .
## Day18               -37.5617098
## Day19                45.0951113
## Day2                -71.6810993
## Day20                 .
## Day21               -20.9653362
## Day22               -11.9218835
## Day23               -15.8115625
## Day24               -92.3942631
## Day25                19.8928737
## Day26               -84.8017992
## Day27                 .
## Day28               -81.4116723
## Day29               -37.0016254
## Day3                 89.5340231
## Day30                 .
## Day31               -35.2859332
## Day4                 -6.2296599
## Day5                  3.3947932
## Day6                 13.0423591
## Day7                 94.8612924
## Day8                  .
## Day9                 19.7400685
## FunctioningYes      959.3879449
```

Relaxed Lasso

```
relax.fit = glmnet(train.x, train.count, relax = TRUE)
relax.lambda = relax.fit$lambda.min
relax.pred = predict(relax.fit, test.x, s = relax.lambda)
mean((relax.pred - test.count)^2)
```

```
## [1] 189335.3
```

Non-Linear Methods

```
#Examine some of the variables in differing degrees of non-linear polynomial functions
fit = lm(Count~ poly(Hour,5)+poly(Temperature,5)+poly(Humidity,5)+poly(Visibility,5)+poly(Solar,5)+poly
summary(fit)
```

```
##
## Call:
## lm(formula = Count ~ poly(Hour, 5) + poly(Temperature, 5) + poly(Humidity,
##     5) + poly(Visibility, 5) + poly(Solar, 5) + poly(Rainfall,
##     5) + Holiday + Month + Day + Functioning, data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1373.84  -206.48   -23.33   167.90  1412.08
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)             -522.484     50.169 -10.414  < 2e-16 ***
## poly(Hour, 5)1         10743.926    406.663  26.420  < 2e-16 ***
## poly(Hour, 5)2          7760.263    753.264  10.302  < 2e-16 ***
## poly(Hour, 5)3         -2116.244    441.944  -4.788 1.73e-06 ***
## poly(Hour, 5)4         -4273.820    464.609  -9.199  < 2e-16 ***
## poly(Hour, 5)5         -8012.762    372.212 -21.527  < 2e-16 ***
## poly(Temperature, 5)1 24045.224   1203.382  19.981  < 2e-16 ***
## poly(Temperature, 5)2  5348.191    559.158   9.565  < 2e-16 ***
## poly(Temperature, 5)3 -5061.365    407.517 -12.420  < 2e-16 ***
## poly(Temperature, 5)4 -4161.305    391.866 -10.619  < 2e-16 ***
## poly(Temperature, 5)5 -1349.479    356.559  -3.785 0.000156 ***
## poly(Humidity, 5)1     -6638.939    647.386 -10.255  < 2e-16 ***
## poly(Humidity, 5)2     -5358.995    427.071 -12.548  < 2e-16 ***
## poly(Humidity, 5)3     -1927.284    387.663  -4.972 6.87e-07 ***
## poly(Humidity, 5)4     -2761.400    361.098  -7.647 2.47e-14 ***
## poly(Humidity, 5)5       202.253    344.120   0.588 0.556734
## poly(Visibility, 5)1     567.218    539.273   1.052 0.292935
## poly(Visibility, 5)2     750.053    375.508   1.997 0.045835 *
## poly(Visibility, 5)3   -1418.907    349.690  -4.058 5.04e-05 ***
## poly(Visibility, 5)4    1401.978    343.362   4.083 4.52e-05 ***
## poly(Visibility, 5)5     -43.491    337.654  -0.129 0.897519
## poly(Solar, 5)1         5821.259    911.567   6.386 1.86e-10 ***
## poly(Solar, 5)2        -5953.504    459.049 -12.969  < 2e-16 ***
## poly(Solar, 5)3         4777.727    388.353  12.303  < 2e-16 ***
## poly(Solar, 5)4        -3120.439    357.177  -8.736  < 2e-16 ***
## poly(Solar, 5)5         2618.301    345.953   7.568 4.51e-14 ***
## poly(Rainfall, 5)1     -2441.856    391.958  -6.230 5.07e-10 ***
## poly(Rainfall, 5)2      1889.200    371.366   5.087 3.77e-07 ***
## poly(Rainfall, 5)3     -1608.831    354.828  -4.534 5.93e-06 ***
## poly(Rainfall, 5)4      2118.304    345.385   6.133 9.31e-10 ***
## poly(Rainfall, 5)5     -1511.674    335.326  -4.508 6.70e-06 ***
## HolidayNo Holiday         56.287     24.342   2.312 0.020803 *
## Month10                  472.492     37.022  12.763  < 2e-16 ***
## Month11                  443.926     29.873  14.860  < 2e-16 ***
## Month12                  126.993     23.465   5.412 6.53e-08 ***
## Month2                    33.499     24.508   1.367 0.171745
## Month3                   211.680     30.533   6.933 4.67e-12 ***
```

```
## Month4                   354.122     35.841   9.880  < 2e-16 ***
## Month5                   350.522     40.060   8.750  < 2e-16 ***
## Month6                   251.779     45.860   5.490 4.22e-08 ***
## Month7                   -74.338     52.062  -1.428 0.153393
## Month8                  -174.662     54.428  -3.209 0.001341 **
## Month9                   221.587     45.730   4.846 1.30e-06 ***
## Day10                      5.380     36.520   0.147 0.882893
## Day11                     65.248     40.303   1.619 0.105527
## Day12                     12.346     38.248   0.323 0.746871
## Day13                     88.273     35.312   2.500 0.012459 *
## Day14                     18.713     36.138   0.518 0.604601
## Day15                    -30.586     36.214  -0.845 0.398385
## Day16                     12.170     36.979   0.329 0.742080
## Day17                    -13.684     36.486  -0.375 0.707633
## Day18                    -22.964     35.758  -0.642 0.520764
## Day19                     59.551     36.505   1.631 0.102887
## Day2                     -60.456     38.596  -1.566 0.117327
## Day20                     26.274     35.008   0.751 0.452982
## Day21                     24.875     37.766   0.659 0.510150
## Day22                     35.277     39.963   0.883 0.377423
## Day23                     -0.814     36.966  -0.022 0.982431
## Day24                    -89.936     37.294  -2.412 0.015921 *
## Day25                    -17.995     34.367  -0.524 0.600576
## Day26                   -104.975     37.075  -2.831 0.004654 **
## Day27                     20.499     36.192   0.566 0.571151
## Day28                    -67.759     35.933  -1.886 0.059393 .
## Day29                    -39.281     37.702  -1.042 0.297522
## Day3                     126.519     41.792   3.027 0.002480 **
## Day30                    -48.737     43.201  -1.128 0.259310
## Day31                      4.024     44.932   0.090 0.928644
## Day4                     -12.447     34.788  -0.358 0.720516
## Day5                     -19.006     36.338  -0.523 0.600986
## Day6                      10.004     34.961   0.286 0.774773
## Day7                      76.523     38.700   1.977 0.048059 *
## Day8                      52.969     39.572   1.339 0.180781
## Day9                      36.000     35.293   1.020 0.307760
## FunctioningYes           994.926     27.316  36.422  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 327.4 on 4790 degrees of freedom
## Multiple R-squared:  0.7186, Adjusted R-squared:  0.7143
## F-statistic: 167.5 on 73 and 4790 DF,  p-value: < 2.2e-16
```

```
#Compare the different functions  on each variable and see which produces the most reliable model
anova1 = gam(Count~Hour+Temperature+Humidity+Visibility+Solar+Rainfall+Holiday+Month+Day+Functioning, da
anova2 = gam(Count~ Hour+s(Temperature,3)+s(Humidity,2)+s(Visibility,2)+s(Solar,3)+Rainfall+Holiday+Mon
anova3 = gam(Count~ Hour+Temperature+s(Humidity,3)+s(Visibility,2)+s(Solar,3)+s(Rainfall,3)+Holiday+Mon
anova4  = gam(Count~ s(Hour,5)+s(Temperature,5)+s(Humidity,4)+s(Solar,4)+s(Rainfall,5)+Holiday+Month+Da
anova(anova1, anova2, anova3, anova4, test = 'F')
```

```
## Analysis of Deviance Table
##
## Model 1: Count ~ Hour + Temperature + Humidity + Visibility + Solar +
```

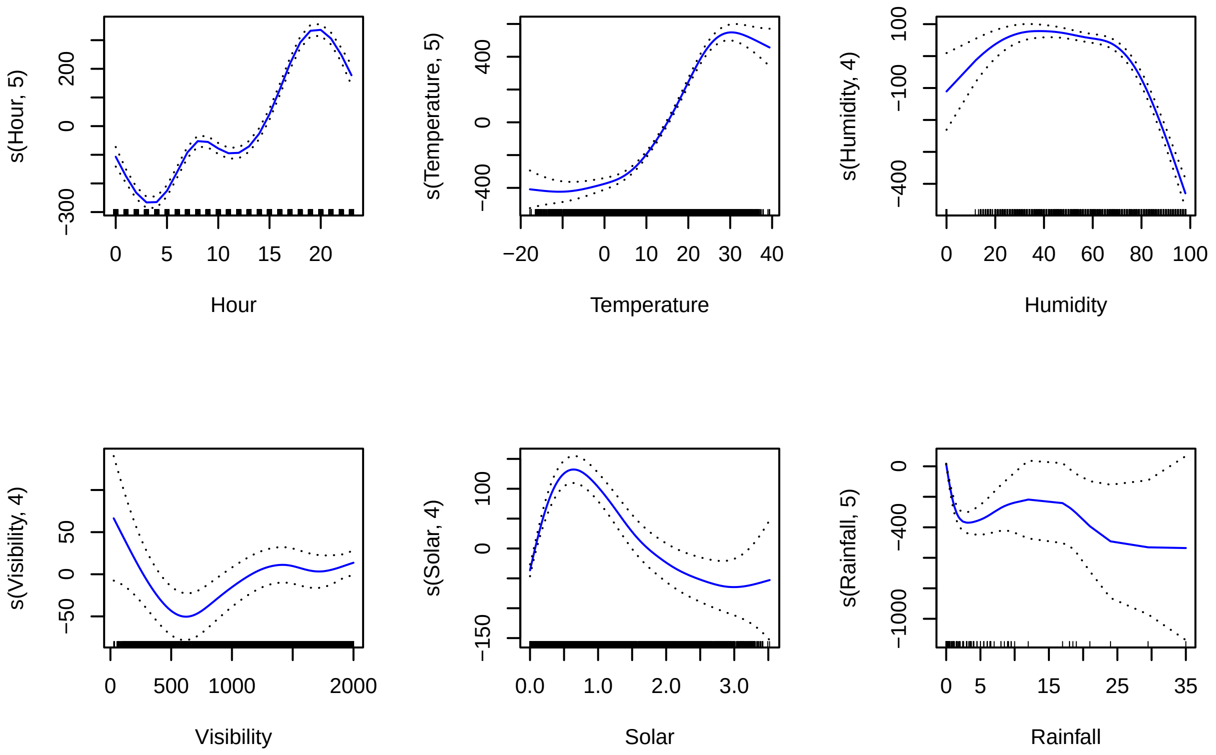
```
##     Rainfall + Holiday + Month + Day + Functioning
## Model 2: Count ~ Hour + s(Temperature, 3) + s(Humidity, 2) + s(Visibility,
##     2) + s(Solar, 3) + Rainfall + Holiday + Month + Day + Functioning
## Model 3: Count ~ Hour + Temperature + s(Humidity, 3) + s(Visibility, 2) +
##     s(Solar, 3) + s(Rainfall, 3) + Holiday + Month + Day + Functioning
## Model 4: Count ~ s(Hour, 5) + s(Temperature, 5) + s(Humidity, 4) + s(Solar,
##     4) + s(Rainfall, 5) + Holiday + Month + Day + Functioning
##   Resid. Df Resid. Dev      Df  Deviance      F    Pr(>F)
## 1      4814  719708622
## 2      4808  646305666 6.00023  73402956 111.60 < 2.2e-16 ***
## 3      4807  653791481 0.99986  -7485815
## 4      4797  525852566 9.99975 127938914 116.71 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
model = glm(Count~ s(Hour,5)+s(Temperature,5)+s(Humidity,4)+s(Solar,4)+s(Rainfall,5)+Holiday+Month+Day+
cv.glm(train, model, K = 10)$delta[1]
```

```
## [1] 152530.3
```

GAM

```
gam.mod = gam(Count~ s(Hour,5)+s(Temperature,5)+s(Humidity,4)+s(Visibility,4)+s(Solar,4)+s(Rainfall,5)+
par(mfrow = c(2,3))
plot(gam.mod, se = TRUE, col = 'blue')
```



```r
summary(gam.mod)
```

```
##
## Call: gam(formula = Count ~ s(Hour, 5) + s(Temperature, 5) + s(Humidity,
##     4) + s(Visibility, 4) + s(Solar, 4) + s(Rainfall, 5) + Holiday +
##     Month + Day + Functioning, data = train)
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -1296.59  -214.78   -33.08   160.09  1452.60
##
## (Dispersion Parameter for gaussian family taken to be 109182.6)
##
##     Null Deviance: 1824319943 on 4863 degrees of freedom
## Residual Deviance: 523312098 on 4793 degrees of freedom
## AIC: 70302.09
##
## Number of Local Scoring Iterations: NA
##
## Anova for Parametric Effects
##                     Df    Sum Sq   Mean Sq   F value    Pr(>F)
## s(Hour, 5)           1 280745339 280745339 2571.3384 < 2.2e-16 ***
## s(Temperature, 5)    1 455217965 455217965 4169.3280 < 2.2e-16 ***
## s(Humidity, 4)       1  50626536  50626536  463.6870 < 2.2e-16 ***
## s(Visibility, 4)     1   3178917   3178917   29.1156 7.147e-08 ***
## s(Solar, 4)          1    349792    349792    3.2037   0.07353 .
## s(Rainfall, 5)       1  10270516  10270516   94.0674 < 2.2e-16 ***
## Holiday              1   1880647   1880647   17.2248 3.378e-05 ***
## Month               11 125090856  11371896  104.1549 < 2.2e-16 ***
## Day                 30  28385828    946194    8.6662 < 2.2e-16 ***
## Functioning          1 145549941 145549941 1333.0877 < 2.2e-16 ***
## Residuals         4793 523312098    109183
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Anova for Nonparametric Effects
##                   Npar Df  Npar F     Pr(F)
## (Intercept)
## s(Hour, 5)              4 173.653 < 2.2e-16 ***
## s(Temperature, 5)       4  90.998 < 2.2e-16 ***
## s(Humidity, 4)          3  96.347 < 2.2e-16 ***
## s(Visibility, 4)        3   6.509 0.0002166 ***
## s(Solar, 4)             3  69.045 < 2.2e-16 ***
## s(Rainfall, 5)          4  35.804 < 2.2e-16 ***
## Holiday
## Month
## Day
## Functioning
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
#Output the R^2 and the MSE
preds = predict(gam.mod, test)
RSS = sum((test$Count - preds)^2)
```

```r
TSS = sum((test$Count - mean(test$Count)) ^ 2)
1 - (RSS / TSS)
```

```
## [1] 0.6945151
```

```r
mean((test$Count - preds)^2)
```

```
## [1] 115668.9
```

K-Nearest Neighbors

```r
set.seed(21)
knn.model = knnreg(train.x, train.count)
knn.pred = predict(knn.model, data.frame(test.x))
mean((test$Count - knn.pred)^2)
```

```
## [1] 189226.8
```

Random Forests

Bagging

```r
bag.fit = randomForest(Count~Hour+Temperature+Humidity+Visibility+Dew+Wind+Snowfall+Solar+Rainfall+Holi
bag.pred = predict(bag.fit, test)
mean((test$Count - bag.pred)^2)
```

```
## [1] 47658.63
```

```r
#Output the variable importance in the bagged random forest model
importance(bag.fit)
```

```
##               %IncMSE IncNodePurity
## Hour        66.326960     489419808
## Temperature 46.138989     540185274
## Humidity    25.149905     176391217
## Visibility  14.514499      38300731
## Dew          7.780979      71410119
## Wind         7.281313      26918076
## Snowfall     5.132555       1082239
## Solar       13.176327     111204486
## Rainfall    14.534428      66212759
## Holiday      5.490983       3798820
## Month        5.671254      59630957
## Day         12.114436      33997108
## Functioning 55.342751     189910776
```

RF

```r
rf.fit = randomForest(Count~Hour+Temperature+Humidity+Visibility+Dew+Wind+Snowfall+Solar+Rainfall+Holida
rf.pred = predict(rf.fit, test)
mean((test$Count - rf.pred)^2)
```

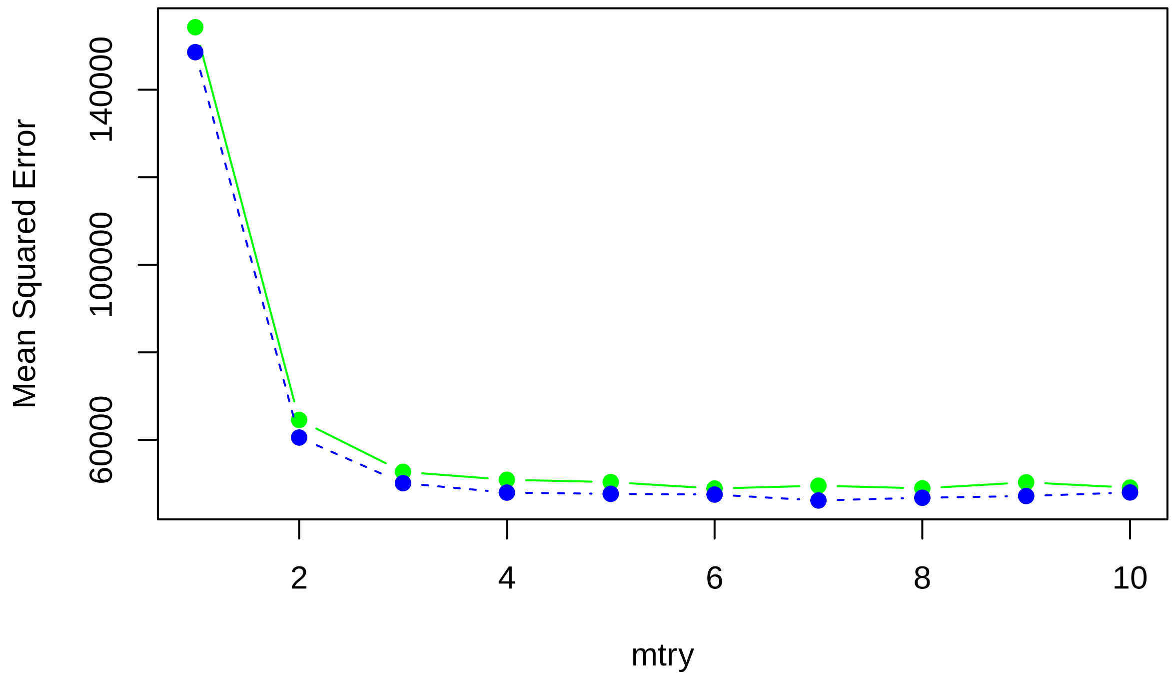
```
## [1] 51281.56
```

```r
importance(rf.fit)
```

```
##               %IncMSE IncNodePurity
## Hour        46.224894     456956560
## Temperature 19.153482     381116213
## Humidity    18.937996     180176308
## Visibility  12.229920      56458951
## Dew          9.259663     150428487
## Wind         7.636612      36426694
## Snowfall     4.064761       3198257
## Solar       12.975940     110319210
## Rainfall    11.555775      66623599
## Holiday      5.946961       3995529
## Month        9.036178     138860325
## Day         13.553250      42258375
## Functioning 34.479912     155387398
```

```r
#Examine how random forest performs with different values of the tuning parameter mtry and see which va
oob.err = double(10)
test.err = double(10)
for(mtry in 1:10)
{
  rf.count = randomForest(Count~Hour+Temperature+Humidity+Dew+Wind+Snowfall+Solar+Rainfall+Holiday+Mont
  oob.err[mtry]=rf.count$mse[50]
  rf.pred = predict(rf.count, test)
  test.err[mtry] = mean((test$Count - rf.pred)^2)
}
```

```r
#Graph the output of the Out-of-Bag error and testing error for each value of mtry
matplot(1:mtry, cbind(test.err, oob.err), pch=19, col=c("green","blue"), type="b", xlab="mtry", ylab="Me
```




```
test.err[7]
```

```
## [1] 49531.73
```

Baseline Test

```
sample_mean = mean(out.data$Count)
mean((test$Count - sample_mean)^2)
```

```
## [1] 378820.8
```

Bagged Estimates

```
#For each of the models selected create new testing data to see how the model performs on average
bag_lin_mse = double(100)
for(i in 1:100)
{
  set.seed(i)
  sampleSize = floor(0.75 * nrow(data))
  split = sample(seq_len(nrow(data)), size = sampleSize)
  new_train = data[split, ]
  new_test = data[-split, ]
  lin.fit = lm(Count~Hour+Temperature+Humidity+Visibility+Solar+Rainfall+Holiday+Month+Day+Functioning,
  lin.pred = predict(lin.fit, new_test, type = 'response')
  bag_lin_mse[i] = mean((lin.pred - new_test$Count)^2)
}
avg = mean(bag_lin_mse)
avg
```

```
## [1] 149123.5
```

```
bag_lasso_mse = double(100)
for(i in 1:100)
{
  set.seed(i)
  sampleSize = floor(0.75 * nrow(data))
  split = sample(seq_len(nrow(data)), size = sampleSize)
  new_train = data[split, ]
  new_test = data[-split, ]
  train.x = model.matrix(Count~Hour+Temperature+Humidity+Visibility+Solar+Rainfall+Holiday+Month+Day+Fu
  train.apps = train$Count
  test.x = model.matrix(Count~Hour+Temperature+Humidity+Visibility+Solar+Rainfall+Holiday+Month+Day+Fun
  test.apps = new_test$Count
  lasso.fit = cv.glmnet(train.x, train.apps, alpha = 1)
  lasso.lambda = lasso.fit$lambda.min
  lasso.pred = predict(lasso.fit, s = lasso.lambda, newx = test.x)
  bag_lasso_mse[i] = mean((lasso.pred - test.apps)^2)
}
avg = mean(bag_lasso_mse)
avg
```

```
## [1] 149350.5
```

```
bag_rf_mse = double(100)
for(i in 1:100)
{
  set.seed(i)
  sampleSize = floor(0.75 * nrow(data))
  split = sample(seq_len(nrow(data)), size = sampleSize)
  new_train = data[split, ]
  new_test = data[-split, ]
  rf.fit = randomForest(Count~Hour+Temperature+Humidity+Visibility+Dew+Wind+Snowfall+Solar+Rainfall+Hol
  rf.pred = predict(rf.fit, new_test)
  bag_rf_mse[i] = mean((new_test$Count - rf.pred)^2)
}
avg = mean(bag_rf_mse)
avg
```

```
## [1] 19799.1
```

```
bag_gam_mse = double(100)
for(i in 1:100)
{
  set.seed(i)
  sampleSize = floor(0.75 * nrow(data))
  split = sample(seq_len(nrow(data)), size = sampleSize)
  new_train = data[split, ]
  new_test = data[-split, ]
  gam.mod = gam(Count~ s(Hour,5)+s(Temperature,5)+s(Humidity,4)+s(Visibility,4)+s(Solar,4)+s(Rainfall,5
  preds = predict(gam.mod, new_test)
  bag_gam_mse[i] = mean((new_test$Count - preds)^2)
}
avg = mean(bag_gam_mse)
avg
```

```
## [1] 108874.7
```

Test Predictions

```r
#Read and convert the data into the proper variable format
test_data = read_csv("~/Documents/STAT1361/test.csv")
```

```
## Rows: 2208 Columns: 14

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (4): Date, Seasons, Holiday, Functioning
## dbl (10): Hour, Temperature, Humidity, Wind, Visibility, Dew, Solar, Rainfal...

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
test_data = test_data %>%
  separate(Date, sep="/", into = c("Day", "Month", "Year"))
```

```r
Count = predict(rf.fit, test_data)
ID = test_data$ID
student_id = rep(4293570, length(Count))
test.pred = data.frame(ID, Count, student_id)
```