Exploratory Data Analysis

Submitted by:

Or Wolfstein

Tom Teman

Jan-Willem Krone

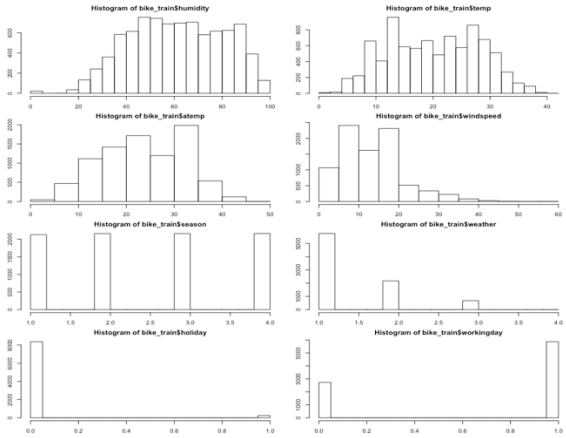
Interdisciplinary Center Herzliya

Global MBA

Instructor: Dr. Roy Sasson

Section One: Descriptive Statistics

First, to get an overview of the data, we have made a histogram for every variable



From these charts, we can see that we have an even distribution of samples in each season, as well we can see that when we look at atemp vs temp, there is a consolidation of samples into fewer bars (this probably means that the "atemp" measurement is not as granular, probably since it is comprised of additional parameters).
The weather is mostly good, and we have only very few holiday samples.

In order to extract more meaningful data, we created a factored hour column from the datetime column:

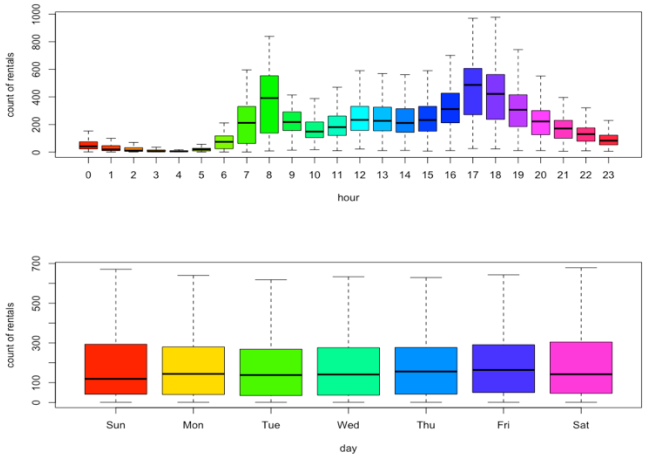
```
bike_train$hour <- as.integer(format(as.POSIXlt(bike_train$datetime), format = "%H"))
bike_train$hour_factored <- as.factor(bike_train$hour)
```

Using the hour column we created a boxplot which shows the count of bike rentals in relation to hours of the day:

```
boxplot(bike_train$count~bike_train$hour_factored,xlab="hour", ylab="count of rentals",
col=rainbow(length(unique(bike_train$hour_factored))))
```

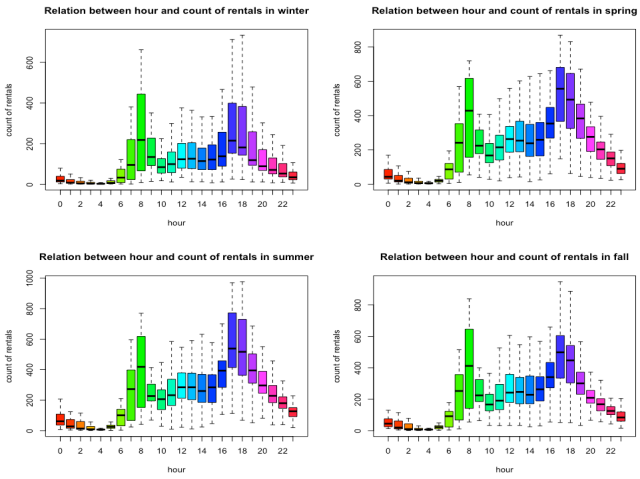
We then created a graph depicting the rental count by day of the week using the R command:

```
boxplot(bike_train$count~bike_train$day_name,xlab="day", ylab="count of rentals",
col=rainbow(length(unique(bike_train$day))),outline=FALSE)
```



We can see that during the early morning hours there is a low volume of bike rentals throughout the entire dataset, the peak hours are in the morning (7-9) and in the evening (16-19). It makes sense as people would use more bikes to drive to and from work.
It is also interesting to see that there is little change between the weekdays and the number of rentals, this chart does not give us any meaningful insights yet

The following charts show the distribution of bike rentals during the day, in each of the seasons.



We see that there is a significant change in the usage of bike rentals between the seasons, the most noticeable being between winter and summer, probably due to the change in weather conditions and temperature.

To further illustrate this point, we plotted out the count of rentals throughout the two years (which allows us to see the different seasons), with the data points colored according to the temperature during each period. We used the R command:

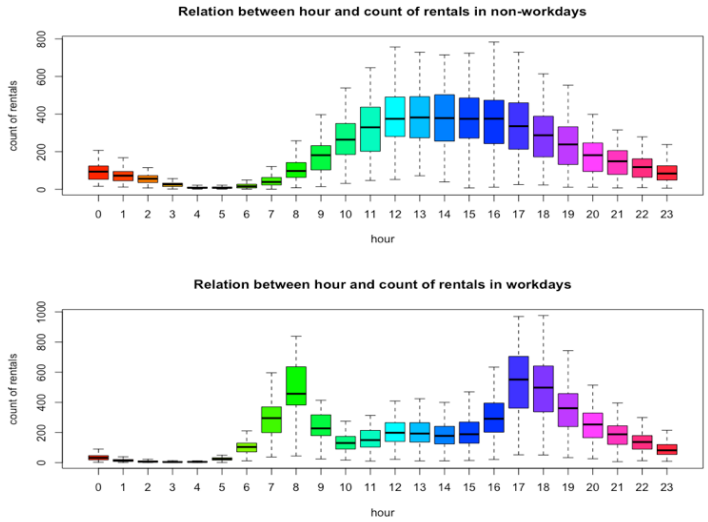
```
pl <- ggplot(bike_train,aes(datetime,count)) + geom_point(aes(color=temp),alpha=0.5)
pl + ggtitle("Relation between count and date, with points colored according to temperature") +
scale_color_continuous(low = '#55D8CE',high = '#FF6E2E') + theme_bw()
```

It then becomes clear to see that the rental count is significantly higher when temperatures are higher as well.

We also wanted to examine the distribution of usage in bike rentals, on work-days VS non-workdays, broken down according to the hours of the day, and plotted the relevant graphs, using the R commands:

```
bike_train_filtered = bike_train[bike_train$workingday == 0, ];
boxplot(main="Relation between hour and count of rentals in non-workdays",
bike_train_filtered$count~bike_train_filtered$hour_factored,xlab="hour", ylab="count of rentals",
col=rainbow(length(unique(bike_train$hour_factored))),outline=FALSE)

bike_train_filtered = bike_train[bike_train$workingday == 1, ];
boxplot(main="Relation between hour and count of rentals in workdays",
bike_train_filtered$count~bike_train_filtered$hour_factored,xlab="hour", ylab="count of rentals",
col=rainbow(length(unique(bike_train_filtered$hour_factored))),outline=FALSE)
```



We can deduce two things out of this: first, it strengths the assumption that users use bikes to get to and from work, and that on non-working days, the volume of usage is also larger, and distributes more evenly throughout the day.

Section Two: Linear Regression

To run the linear regression, we ran the following commands in R:

```
bikes_train_lm <- lm(data = bike_train, count ~ temp)
summary(bikes_train_lm)
```

Which yielded:

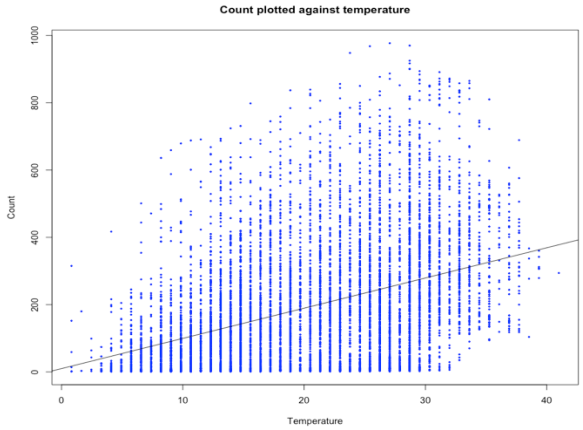
```
Residuals:
    Min      1Q  Median      3Q     Max
-291.03 -109.79  -33.13   77.60  729.87

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   9.8601     4.8367   2.039   0.0415 *
temp          8.9798     0.2234  40.199   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 165.7 on 8598 degrees of freedom
Multiple R-squared:  0.1582,    Adjusted R-squared:  0.1581
F-statistic:  1616 on 1 and 8598 DF,  p-value: < 2.2e-16
```

We then drew the plot with the regression line on it using the following commands in R:

```
plot(bike_train$temp, bike_train$count, pch = 20, cex = .5, col = "blue", main = "Count plotted
against temperature", xlab = "Temperature", ylab = "Count")
abline(lm(data = bike_train, count ~ temp))
```



We then divided the data into two subsets - train and test using the following R commands:

```
# 70% of the sample size
smp_size <- floor(0.7 * nrow(bike_train))
set.seed(4242)
train_ind <- sample(seq_len(nrow(bike_train)), size = smp_size)
subset_train <- bike_train[train_ind, ]
subset_test <- bike_train[-train_ind, ]
```

We then estimated a linear model with count as a dependent variable and temp and hour as the independent variables using the following R command:

```
subset_train_lm <- lm(data = subset_train, count ~ temp + hour)
summary(subset_train_lm)
```

Which yielded the following results:

```
Residuals:
    Min      1Q  Median      3Q     Max
-311.89 -101.76  -31.35   59.92  677.16

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -73.6332     6.0090  -12.25   <2e-16 ***
temp          7.9081     0.2502   31.61   <2e-16 ***
hour          9.2009     0.2916   31.55   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 154.5 on 6017 degrees of freedom
Multiple R-squared:  0.2772,    Adjusted R-squared:  0.2769
F-statistic:  1154 on 2 and 6017 DF,  p-value: < 2.2e-16
```

The hour coefficients don't make too much sense since unlike temperature, our variables are made up of discrete integers and have no continuous values. Categorizing its values into factors will provide higher accuracy in model building. So we ran the same model, this time treating the hour variable as a factor:

```
subset_train_lm_factored <- lm(data = subset_train, count ~ temp + hour_factored)
summary(subset_train_lm_factored)
```

Which yielded the following results:

```
Residuals:
    Min      1Q  Median      3Q     Max
-396.96  -62.74   -6.25   51.80  508.38

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)     -68.7277     8.1280  -8.456  < 2e-16 ***
temp              6.5077     0.1908  34.099  < 2e-16 ***
hour_factored1  -15.6240    10.3535  -1.509 0.131338
hour_factored2  -27.9296    10.2913  -2.714 0.006669 **
hour_factored3  -38.1321    10.6737  -3.573 0.000356 ***
hour_factored4  -41.7734    10.3338  -4.042 5.36e-05 ***
hour_factored5  -25.3757    10.2048  -2.487 0.012922 *
hour_factored6   33.6960    10.2546   3.286 0.001022 **
hour_factored7  162.4003    10.1550  15.992  < 2e-16 ***
hour_factored8  313.6665    10.2702  30.541  < 2e-16 ***
```

```
hour_factored9  166.0975    10.3747  16.010  < 2e-16 ***
hour_factored10 107.2754    10.2935  10.422  < 2e-16 ***
hour_factored11 139.0408    10.2291  13.593  < 2e-16 ***
hour_factored12 180.6996    10.2551  17.620  < 2e-16 ***
hour_factored13 179.2455    10.1943  17.583  < 2e-16 ***
hour_factored14 160.8549    10.3287  15.574  < 2e-16 ***
hour_factored15 162.7229    10.3875  15.665  < 2e-16 ***
hour_factored16 236.5180    10.2809  23.006  < 2e-16 ***
hour_factored17 393.4812    10.2645  38.334  < 2e-16 ***
hour_factored18 362.9259    10.2483  35.413  < 2e-16 ***
hour_factored19 242.5423    10.4010  23.319  < 2e-16 ***
hour_factored20 161.6712    10.2850  15.719  < 2e-16 ***
hour_factored21 111.3035    10.3344  10.770  < 2e-16 ***
hour_factored22  72.4204    10.4428   6.935 4.49e-12 ***
hour_factored23  35.6259    10.1726   3.502 0.000465 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 115.3 on 5995 degrees of freedom
Multiple R-squared:  0.5988,    Adjusted R-squared:  0.5972
F-statistic: 372.9 on 24 and 5995 DF,  p-value: < 2.2e-16
```

Treating hour as a factored variable allows us to investigate the correlation between count and interesting time windows during the day (for instance, between 16:00 and 19:00, as we witnessed in our initial analysis). In order to have better predictions, we will create a new categorical variable of "period in the day" by using the ifelse function. We split up the day into 5 time windows: night (23:00-06:00), morning commute (07:00-09:00), midday (10:00-16:00), evening commute (17:00-19:00) and late evening (20:00-22:00).

```
bike_train$hour_window<-NA
bike_train$hour_window<-ifelse(bike_train$hour>=0 & bike_train$hour<=6 |
bike_train$hour==23,"night", bike_train$hour_window)
bike_train$hour_window<-ifelse(bike_train$hour>=7 & bike_train$hour<=9,"morning commute",
bike_train$hour_window)
bike_train$hour_window<-ifelse(bike_train$hour>=10 & bike_train$hour<=15,"midday",
bike_train$hour_window)
bike_train$hour_window<-ifelse(bike_train$hour>=16 & bike_train$hour<=19,"evening commute",
bike_train$hour_window)
bike_train$hour_window<-ifelse(bike_train$hour>=20 & bike_train$hour<=22,"late evening",
bike_train$hour_window)
bike_train$hour_window <- as.factor(bike_train$hour_window)
```

In addition, to get clearer results from the linear regression model, we releveled 'midday' to be the baseline value (this seemed to make to most sense to us due to the values we saw in the initial inspection of the data). We then ran a linear regression using the following R commands:

```
subset_train$hour_window=relevel(subset_train$hour_window, "midday")

subset_train.lm <- lm(data = subset_train, count ~ temp + hour_window)
summary(subset_train.lm)
```

Which yielded the following results:

```
Residuals:
    Min      1Q  Median      3Q     Max
-355.27  -75.05  -10.57   53.48  560.73

Coefficients:
                          Estimate Std. Error t value Pr(>|t|)
(Intercept)                82.4086     5.5191  14.932  < 2e-16 ***
temp                        6.6891     0.2038  32.818  < 2e-16 ***
hour_windowevening commute 154.8209     5.0201  30.840  < 2e-16 ***
hour_windowlate evening    -38.7802     5.5659  -6.967 3.57e-12 ***
hour_windowmorning commute  59.3648     5.5389  10.718  < 2e-16 ***
hour_windownight          -163.5289     4.2888 -38.129  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 123.7 on 6014 degrees of freedom
Multiple R-squared:  0.5367,    Adjusted R-squared:  0.5363
F-statistic:  1393 on 5 and 6014 DF,  p-value: < 2.2e-16
```

In which it is clear that 'evening commute' and 'morning commute' have the highest positive deviation in regards to 'count' as opposed to the other time windows (when 'midday', the "average" value is the reference point). The adjusted R-squared is slightly above 50%, which means that the

model is quite weak as a predictive model, but indicates that these variables (temp and hour_window) are meaningful predictors.

To further investigate their value as predictors, we have estimated another linear regression model, this time introducing an interaction between temp and hour_window:

```
subset_train.im <- lm(data = subset_train, count ~ temp*hour_window)
summary(subset_train.im)
```

Which yielded the following results:

```
Residuals:
    Min      1Q  Median      3Q     Max
-400.77  -56.95  -17.44   48.14  561.56

Coefficients:
                               Estimate Std. Error t value Pr(>|t|)
(Intercept)                     73.8536     8.7819   8.410  < 2e-16 ***
temp                             7.0750     0.3713  19.056  < 2e-16 ***
hour_windowevening commute       4.8472    14.1242   0.343   0.7315
hour_windowlate evening        -70.7026    15.4072  -4.589 4.55e-06 ***
hour_windowmorning commute      81.9856    14.0494   5.836 5.64e-09 ***
hour_windownight               -64.2611    11.2728  -5.701 1.25e-08 ***
temp:hour_windowevening commute  6.7658     0.5984  11.306  < 2e-16 ***
temp:hour_windowlate evening     1.6136     0.6940   2.325   0.0201 *
temp:hour_windowmorning commute -1.1431     0.6573  -1.739   0.0821 .
temp:hour_windownight           -5.3801     0.5172 -10.402  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 119.4 on 6010 degrees of freedom
Multiple R-squared:  0.5683,    Adjusted R-squared:  0.5676
F-statistic: 878.9 on 9 and 6010 DF,  p-value: < 2.2e-16
```

Basically, including temp*hour_window in the model formula means we're fitting temp, hour_window, and temp:hour_window (the interaction of temp and hour_window). The interaction term is statistically significant, suggesting that the effect of the hour_window is different for each of the temperatures.

The adjusted R-squared compares the explanatory power of regression models that contain different numbers of predictors. It is a modified version of R-squared that has been adjusted for the number of predictors in the model. It increases only if the new term improves the model more than would be expected by chance. It decreases when a predictor improves the model by less than expected by chance.

In our case, comparing the value of adjusted R-squared of these two models (0.5363 vs. 0.5676) indicates that the additional term of interaction between temp and hour_window improves the model by more than pure chance, and indicates that it is indeed better for prediction. It also ties in with our initial analysis, which indicated a correlation between the time of day and the number of bike rentals.

To test which model offers a better prediction, we ran the prediction for each model generated from our training data against the test data (both are subsets of the original bike_train data), and computed the R-squared value from both prediction results, using the following R commands:

```
subset_test$predictTest = predict(subset_train.lm, subset_test)
lm.sse_test = sum((subset_test$count - subset_test$predictTest)^2)
lm.sst_test = sum((subset_test$count - mean(bike_train$count))^2)
1 - lm.sse_test/lm.sst_test

subset_test$predictTest2 = predict(subset_train.im, subset_test)
im.sse_test = sum((subset_test$count - subset_test$predictTest2)^2)
im.sst_test = sum((subset_test$count - mean(bike_train$count))^2)
1 - im.sse_test/im.sst_test
```

Which yielded 0.5302129 for the simple linear model and 0.5611122 for the interaction linear model. In our case, a higher R-squared value does indicate a better predictor, but in general a higher

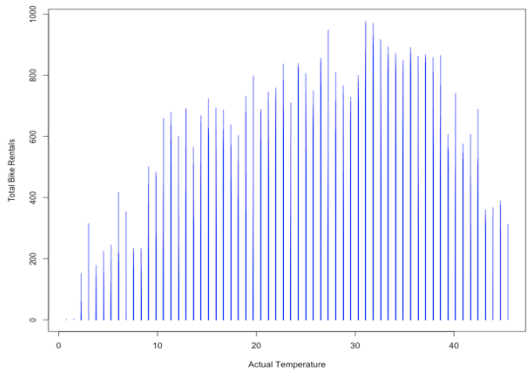
R-squared value doesn't necessarily mean a better fit since you can always excessively bend the fitted line to artificially connect the dots rather than finding a true relationship between the variables. In our case we don't believe we conducted any over-fitting, hence, the better predictor is the interaction linear model.

Section 3: Final Model

We began by finding out that the weather variables with the highest correlation to the count variable are humidity (negative correlation of around -0.32) and atemp/temp (each with a positive correlation of about 0.37).
We decided to use atemp moving forward since it encapsulates both humidity and windspeed. Therefore, we decided the aggregated weather variable will consist of these two continuous values (humidity and atemp) as well as the factored weather variable. The season variable seemed irrelevant to us, since it takes form in the weather variables themselves (i.e. in winter you get low temperatures and in summer they are high).

While the humidity value is already normalized between 0 and 100, the actual temperature variable isn't. We wanted to find out which atemp value is optimal, so we plotted the rental count as a function of atemp using the following R command:

```
plot(bike_train$atemp, bike_train$count, type = 'h', col= 'blue', xlab = 'Actual Temperature', ylab
= 'Total Bike Rentals')
```



From this we saw that at a temperature of about 32 degrees Celsius, the amount of bike rentals peaks. Therefore, we created a new variable, called atemp_normalized, which was calculated in the following manner:

```
bike_train$atemp_normalized<-ifelse(bike_train$atemp>32, 60+(46-bike_train$atemp)*40/14,
bike_train$atemp_normalized);
bike_train$atemp_normalized<-ifelse(bike_train$atemp<=32, bike_train$atemp*100/32,
bike_train$atemp_normalized);
```

Explained: if the atemp is below 32 degrees, normalize the value between 0 and 100 (where 32 degrees is 100 and 0 degrees is 0), and if the value is above 32 degrees, normalize the value between 60 and 100 (where 32 degrees is 100 and 46 degrees, the highest atemp value, is 60).

We then moved on to create the aggregated weather variable, which was done using the following formula:

```
bike_train$agg_temp = (bike_train$atemp_normalized - bike_train$humidity)*4 - bike_train$weather*10
+ bike_train$season*25
```

The values of 4, 10 and 25 were reached through trial and error, striving to reach the highest correlation value between agg_temp and count (0.5075025).

We then generated a linear regression model, making sure to interact heavily with the hour_window variable:

```
bike_train.final <- lm(data = bike_train, count ~
holiday_factored*hour_window+workingday_factored*hour_window+hour*hour_window+agg_temp*hour_window)
summary(bike_train.final)
```

Which yielded the following results:

```
Residuals:
    Min      1Q  Median      3Q     Max
-502.49  -41.91   -9.31   39.08  483.14

Coefficients:
                                                          Estimate Std. Error t value Pr(>|t|)
(Intercept)                                               -9.04328   40.81741  -0.222 0.824666
holiday_factoredHoliday                                   36.59822   15.93486   2.297 0.021658 *
hour_windowlate evening                                  932.03293   86.40828  10.786  < 2e-16 ***
hour_windowmidday                                        229.31480   43.73290   5.244 1.61e-07 ***
hour_windowmorning commute                               139.45017   50.31597   2.771 0.005592 **
hour_windownight                                          39.95265   40.98074   0.975 0.329631
workingday_factoredWorking_Day                            95.67560    5.64090  16.961  < 2e-16 ***
hour                                                      10.24006    2.29003   4.472 7.86e-06 ***
agg_temp                                                   1.02845    0.02146  47.917  < 2e-16 ***
holiday_factoredHoliday:hour_windowlate evening           10.28189   24.34128   0.422 0.672740
holiday_factoredHoliday:hour_windowmidday                -81.38668   20.57999  -3.955 7.73e-05 ***
holiday_factoredHoliday:hour_windowmorning commute        16.38826   24.39360   0.672 0.501712
holiday_factoredHoliday:hour_windownight                 -49.95573   19.56568  -2.553 0.010690 *
hour_windowlate evening:workingday_factoredWorking_Day   -61.69900    8.61364  -7.163 8.55e-13 ***
hour_windowmidday:workingday_factoredWorking_Day        -281.79765    7.28002 -38.708  < 2e-16 ***
hour_windowmorning commute:workingday_factoredWorking_Day 128.61823   8.61138  14.936  < 2e-16 ***
hour_windownight:workingday_factoredWorking_Day         -108.51091    6.91645 -15.689  < 2e-16 ***
hour_windowlate evening:hour                             -49.18352    4.27215 -11.513  < 2e-16 ***
hour_windowmidday:hour                                    -5.50143    2.60663  -2.111 0.034840 *
hour_windowmorning commute:hour                          -14.80723    4.28671  -3.454 0.000555 ***
hour_windownight:hour                                     -7.66139    2.30494  -3.324 0.000891 ***
hour_windowlate evening:agg_temp                          -0.47037    0.03368 -13.965  < 2e-16 ***
hour_windowmidday:agg_temp                                -0.41292    0.02785 -14.824  < 2e-16 ***
hour_windowmorning commute:agg_temp                       -0.58495    0.03275 -17.860  < 2e-16 ***
hour_windownight:agg_temp                                 -0.90784    0.02726 -33.305  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 96.26 on 8575 degrees of freedom
Multiple R-squared:  0.7168,    Adjusted R-squared:  0.716
F-statistic: 904.4 on 24 and 8575 DF,  p-value: < 2.2e-16
```

The value of the adjusted R-squared is 0.716, which indicates that this is a good linear model and we feel confident moving forward and using it on bike_test.
As mentioned above, we place a heavy emphasis on hour_window, as our preliminary analysis of the data indicated that it has a large influence over the number of bike rentals, hence probably a strong correlation. We differentiated between holidays and working days. In addition, we added the interaction between hour and hour window, to make sure to account for the subtle changes within the windows themselves. And obviously, we included an interaction between our agg_temp variable and hour_window, since we believe them to be the most influential parameters.

Therefore, we create our predictor for the test file using the following R command:

```
subset_test$predictTest = predict(bike_train.final, subset_test)
```

And add the generated 'count' values to the bike_test dataset:

```
bike_test$count = floor(predict(bike_train.final, bike_test))
```

And also clean it up from any outliers before writing it to the file:

```
bike_test$count = ifelse(bike_test$count < 0, 0, bike_test$count)
write.csv(bike_test, file = "bike_test.csv")
```

As required, we manually calculate the 'count' value for two random rows in bike_test. The formula, generated using the coefficients from the linear model, is:

```
count ~ -9.04   +
        36.6    * holiday_factoredHoliday +
        932.03  * hour_windowlate_evening +
        229.31  * hour_windowmidday +
        139.45  * hour_windowmorning_commute +
        39.95   * hour_windownight +
        95.68   * workingday_factoredWorking_Day +
        10.24   * hour +
```

```
        1.03    * agg_temp +
        10.28   * holiday_factoredHoliday * hour_windowlate_evening +
        -81.39  * holiday_factoredHoliday * hour_windowmidday +
        16.39   * holiday_factoredHoliday * hour_windowmorning_commute +
        -49.96  * holiday_factoredHoliday * hour_windownight +
        -61.7   * hour_windowlate_evening * workingday_factoredWorking_Day +
        -281.8  * hour_windowmidday * workingday_factoredWorking_Day +
        128.62  * hour_windowmorning_commute * workingday_factoredWorking_Day +
        -108.51 * hour_windownight * workingday_factoredWorking_Day +
        -49.18  * hour_windowlate_evening * hour +
        -5.5    * hour_windowmidday * hour +
        -14.81  * hour_windowmorning_commute * hour +
        -7.66   * hour_windownight * hour +
        -0.47   * hour_windowlate_evening * agg_temp +
        -0.41   * hour_windowmidday * agg_temp +
        -0.58   * hour_windowmorning_commute * agg_temp +
        -0.91   * hour_windownight * agg_temp
```

Plugging in the values from two random rows we get the following results:

Row 42:

```
count ~ -9.04   +
        95.68   * 1 (workingday_factoredWorking_Day) +
        10.24   * 18 (hour) +
        1.03    * 110.6875 (agg_temp) =
        384.789959375
```

And if you floor the value like we did, you get 384, which is indeed the predicted count value in row 42.

Row 1336:

```
count ~ -9.04   +
        139.45  * 1 (hour_windowmorning_commute) +
        10.24   * 7 (hour) +
        1.03    * 61.0625 (agg_temp) +
        -14.81  * 1 (hour_windowmorning_commute) * 7 (hour) +
        -0.58   * 1 (hour_windowmorning_commute) * 61.0625 (agg_temp) =
        125.898125
```

And if you floor the value like we did, you get 125, which is indeed the predicted count value in row 1336.