## Model Validation Assignment

Needed libraries

tidyverse.quiet = TRUE  
library(tidyverse)  
library(caret)  
library(MASS)  
library(ggcorrplot)

Reading Data

bike <- read.csv("hour.csv")  
bike = bike %>% drop\_na() #delete any row with an NA value

bike = bike %>% mutate(season = as\_factor(as.character(season))) %>%  
mutate(season = fct\_recode(season,  
"Spring" = "1",  
"Summer" = "2",  
"Fall" = "3",  
"Winter" = "4"))

bike = bike %>% mutate(yr = as\_factor(yr)) %>% mutate(mnth = as\_factor(mnth)) %>%  
 mutate(hr = as\_factor(hr))   
  
bike = bike %>% mutate(holiday = as\_factor(holiday)) %>%  
mutate(holiday = fct\_recode(holiday,  
"NotHoliday" = "0",  
"Holiday" = "1"))  
  
bike = bike %>% mutate(workingday = as\_factor(workingday)) %>%  
mutate(workingday = fct\_recode(workingday,  
"NotWorkingDay" = "0",  
"WorkingDay" = "1"))  
  
bike = bike %>% mutate(weathersit = as\_factor(weathersit)) %>%  
mutate(weathersit = fct\_recode(weathersit,  
"NoPrecip" = "1",  
"Misty" = "2",  
"LightPrecip" = "3",  
"HeavyPrecip" = "4"))  
  
bike = bike %>% mutate(weekday = as\_factor(weekday)) %>%  
mutate(weekday = fct\_recode(weekday,  
"Sunday" = "0",   
"Monday" = "1",  
"Tuesday" = "2",  
"Wednesday" = "3",  
"Thursday" = "4",  
"Friday" = "5",  
"Saturday" = "6"))

bike2 <- bike %>% dplyr::select(-c(instant, dteday, registered, casual, yr, workingday, atemp, hum, windspeed))

Split the data (training and testing)

set.seed(1234)  
train.rows = createDataPartition(y = bike2$count, p=0.7, list = FALSE) #70% in training  
train = slice(bike2,train.rows)   
test = slice(bike2,-train.rows)

We have 12167 rows in the training set and 5212 rows in the test set.

Model with multiple variables.

#create linear regression model  
mod\_train = lm(count ~ season + mnth + hr + holiday + weekday + holiday + temp + weathersit, data = train)   
summary(mod\_train) #examine the model

##   
## Call:  
## lm(formula = count ~ season + mnth + hr + holiday + weekday +   
## holiday + temp + weathersit, data = train)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -419.31 -61.93 -9.98 52.57 504.24   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -95.3295 6.8961 -13.824 < 2e-16 \*\*\*  
## seasonSummer 28.8486 6.4074 4.502 6.78e-06 \*\*\*  
## seasonFall 19.7865 7.6029 2.602 0.009266 \*\*   
## seasonWinter 62.0339 6.4333 9.643 < 2e-16 \*\*\*  
## mnth2 -0.8013 5.1396 -0.156 0.876114   
## mnth3 2.5584 5.7973 0.441 0.659003   
## mnth4 -1.2250 8.6334 -0.142 0.887166   
## mnth5 -1.5879 9.2279 -0.172 0.863382   
## mnth6 -15.3992 9.4846 -1.624 0.104485   
## mnth7 -38.8277 10.6085 -3.660 0.000253 \*\*\*  
## mnth8 -16.8557 10.3542 -1.628 0.103569   
## mnth9 5.4060 9.2152 0.587 0.557459   
## mnth10 -2.7341 8.5079 -0.321 0.747943   
## mnth11 -12.8043 8.2169 -1.558 0.119193   
## mnth12 -15.3615 6.5409 -2.349 0.018864 \*   
## hr1 -19.7855 6.9722 -2.838 0.004550 \*\*   
## hr2 -28.2440 6.9696 -4.052 5.10e-05 \*\*\*  
## hr3 -40.3146 7.0910 -5.685 1.34e-08 \*\*\*  
## hr4 -40.5469 7.0249 -5.772 8.03e-09 \*\*\*  
## hr5 -26.7454 6.9592 -3.843 0.000122 \*\*\*  
## hr6 32.8518 7.0435 4.664 3.13e-06 \*\*\*  
## hr7 161.3872 6.9925 23.080 < 2e-16 \*\*\*  
## hr8 312.2263 6.9502 44.923 < 2e-16 \*\*\*  
## hr9 164.2556 7.0163 23.411 < 2e-16 \*\*\*  
## hr10 107.1856 6.9552 15.411 < 2e-16 \*\*\*  
## hr11 139.6256 7.0057 19.930 < 2e-16 \*\*\*  
## hr12 179.7448 6.9778 25.760 < 2e-16 \*\*\*  
## hr13 178.6812 7.0201 25.453 < 2e-16 \*\*\*  
## hr14 156.2811 7.0628 22.127 < 2e-16 \*\*\*  
## hr15 168.7543 7.0939 23.788 < 2e-16 \*\*\*  
## hr16 228.1106 7.0881 32.182 < 2e-16 \*\*\*  
## hr17 377.6085 7.0185 53.802 < 2e-16 \*\*\*  
## hr18 347.7287 6.9806 49.813 < 2e-16 \*\*\*  
## hr19 238.7339 7.0128 34.043 < 2e-16 \*\*\*  
## hr20 159.7394 7.0231 22.745 < 2e-16 \*\*\*  
## hr21 108.1070 6.9494 15.556 < 2e-16 \*\*\*  
## hr22 72.3808 6.9874 10.359 < 2e-16 \*\*\*  
## hr23 32.5734 6.9996 4.654 3.30e-06 \*\*\*  
## holidayHoliday -29.0249 6.4088 -4.529 5.98e-06 \*\*\*  
## weekdayMonday 7.5047 3.8928 1.928 0.053894 .   
## weekdayTuesday 6.7559 3.8314 1.763 0.077878 .   
## weekdayWednesday 10.7642 3.7993 2.833 0.004617 \*\*   
## weekdayThursday 12.3082 3.8052 3.235 0.001221 \*\*   
## weekdayFriday 15.3600 3.7730 4.071 4.71e-05 \*\*\*  
## weekdaySaturday 14.0349 3.7638 3.729 0.000193 \*\*\*  
## temp 288.1743 12.1860 23.648 < 2e-16 \*\*\*  
## weathersitMisty -19.6696 2.3717 -8.293 < 2e-16 \*\*\*  
## weathersitLightPrecip -94.1331 3.8166 -24.664 < 2e-16 \*\*\*  
## weathersitHeavyPrecip -80.2490 64.7672 -1.239 0.215356   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 111.9 on 12118 degrees of freedom  
## Multiple R-squared: 0.6217, Adjusted R-squared: 0.6202   
## F-statistic: 414.8 on 48 and 12118 DF, p-value: < 2.2e-16

All the variables used in the regression model has atleast one factor that is significant. The R-Squared value at .6202 is good. The residuals based on the minimum and maximum seem spread out.

predict\_train = predict(mod\_train, newdata = train, interval = "predict", type = "response")  
  
head(predict\_train)

## fit lwr upr  
## 1 -37.68169 -257.3449 181.9815  
## 2 -46.14026 -265.8041 173.5236  
## 3 -52.44730 -272.1354 167.2408  
## 4 -52.67962 -272.3518 166.9925  
## 5 -58.54772 -278.2314 161.1359  
## 6 14.95557 -204.7171 234.6282

summary(predict\_train)

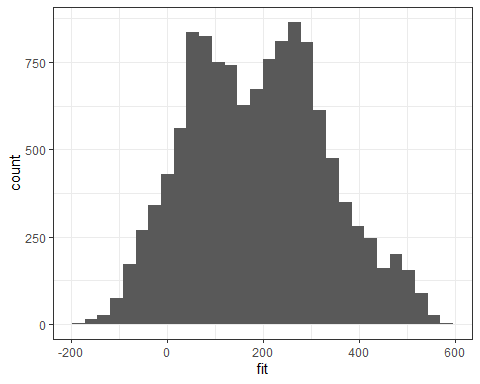
## fit lwr upr   
## Min. :-183.90 Min. :-403.67 Min. : 35.87   
## 1st Qu.: 75.04 1st Qu.:-144.72 1st Qu.:294.74   
## Median : 188.25 Median : -31.42 Median :407.95   
## Mean : 189.33 Mean : -30.40 Mean :409.06   
## 3rd Qu.: 289.19 3rd Qu.: 69.33 3rd Qu.:508.92   
## Max. : 584.44 Max. : 364.57 Max. :804.31

temp\_var = predict(mod\_train, interval = "prediction")

## Warning in predict.lm(mod\_train, interval = "prediction"): predictions on current data refer to \_future\_ responses

train\_df = cbind(train, temp\_var)  
  
ggplot(train\_df, aes(x = fit)) +   
 geom\_histogram() +  
 theme\_bw()

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.



The predictions have negative values. The histogram of the predictions on the training set is normal, hence reasonable.

predict\_test = predict(mod\_train, newdata = test, interval = "predict", type = "response")  
  
head(predict\_test)

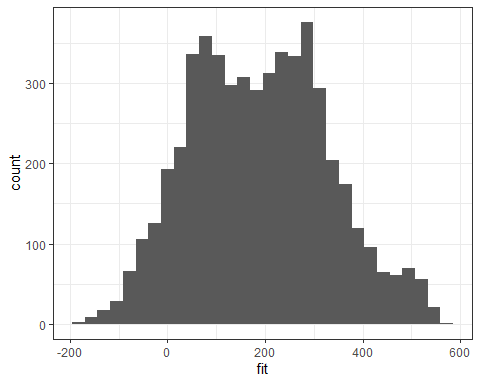
## fit lwr upr  
## 1 -12.13272 -231.79110 207.5257  
## 2 137.72755 -81.94175 357.3968  
## 3 174.04493 -45.67523 393.7651  
## 4 17.56108 -202.19963 237.3218  
## 5 -22.20993 -241.96034 197.5405  
## 6 168.48847 -51.19453 388.1715

summary(predict\_test)

## fit lwr upr   
## Min. :-187.00 Min. :-406.80 Min. : 32.81   
## 1st Qu.: 78.66 1st Qu.:-141.01 1st Qu.:298.36   
## Median : 188.34 Median : -31.41 Median :408.07   
## Mean : 189.41 Mean : -30.31 Mean :409.13   
## 3rd Qu.: 288.72 3rd Qu.: 69.01 3rd Qu.:508.43   
## Max. : 567.39 Max. : 347.64 Max. :787.14

temp\_var2 = predict(mod\_train, newdata = test, interval = "prediction")  
test\_df = cbind(test, temp\_var2)  
  
  
ggplot(test\_df, aes(x = fit)) +   
 geom\_histogram() +  
 theme\_bw()

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.

 The predictions have more positive values as it should be. The histogram of the predictions on the test set is normal and very similar to the one based on training set, hence reasonable.

Now we can manually calculate the R squared value.

SSE = sum((test$count - predict\_test)^2) #sum of squared residuals from model  
SST = sum((test$count - mean(test$count))^2) #sum of squared residuals from a "naive" model  
1 - SSE/SST #definition of R squared

## [1] -3.058344

The R squared value of 0.6202 on the training set is close enough to the manually calculated R squared value of 0.6289 on the test set.

K-fold cross-validation; splits the data into n sets of equal rows(ex. say 5), then it uses n-1 sets for training and the one set left out as test, this is repeated until each set is used as a test set. Training/split uses all data similar to k-fold, but once based on the split defined by the model 70/30 percent split for example.