# Module 2 Assignment 2

## Maliszewski, Angela

## Multiple Linear Regression and Special Issues

Libraries: For this assignment you may need the following libraries: tidyverse, tidymodels, *glmnet*, GGally, ggcorrplot, MASS, *car*, lubridate, and lmtest. Feel free to install and library any other packages that you feel are needed.

library(tidyverse)

## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.3 v purrr 0.3.4  
## v tibble 3.0.5 v dplyr 1.0.3  
## v tidyr 1.1.2 v stringr 1.4.0  
## v readr 1.3.1 v forcats 0.5.0

## -- Conflicts ------------------------------------------ tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(tidymodels)

## -- Attaching packages -------------------------------------- tidymodels 0.1.2 --

## v broom 0.7.4 v recipes 0.1.15  
## v dials 0.0.9 v rsample 0.0.8   
## v infer 0.5.4 v tune 0.1.2   
## v modeldata 0.1.0 v workflows 0.2.1   
## v parsnip 0.1.5 v yardstick 0.0.7

## -- Conflicts ----------------------------------------- tidymodels\_conflicts() --  
## x scales::discard() masks purrr::discard()  
## x dplyr::filter() masks stats::filter()  
## x recipes::fixed() masks stringr::fixed()  
## x dplyr::lag() masks stats::lag()  
## x yardstick::spec() masks readr::spec()  
## x recipes::step() masks stats::step()

library(GGally)

## Registered S3 method overwritten by 'GGally':  
## method from   
## +.gg ggplot2

library(ggcorrplot)  
library(MASS)

##   
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':  
##   
## select

library(lubridate)

##   
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':  
##   
## date, intersect, setdiff, union

library(lmtest)

## Loading required package: zoo

##   
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':  
##   
## as.Date, as.Date.numeric

library(dplyr)  
library(readr)

#install.packages("glmnet")  
library(glmnet)

## Loading required package: Matrix

##   
## Attaching package: 'Matrix'

## The following objects are masked from 'package:tidyr':  
##   
## expand, pack, unpack

## Loaded glmnet 4.1

#install.packages("car")  
library(car)

## Loading required package: carData

##   
## Attaching package: 'car'

## The following object is masked from 'package:dplyr':  
##   
## recode

## The following object is masked from 'package:purrr':  
##   
## some

#Task 1: Read in the data from the “bike\_cleaned.csv” file into a data frame/tibble named “bike”. Several of the variables need to be converted into correct types before we can proceed: Convert “dteday” from a character variable to a date variable. The code below will perform this conversion:

bike = bike %>% mutate(dteday = mdy(dteday)) #mdy is a lubridate package function

Convert the remaining character variables to factors. You can do this one variable at a time or use a “mutate\_if”.

Finally, convert the “hr” variable into a factor. Why do we convert the “hr” variable into factor? Why not just leave as numbers?

ANSWER: Converting hr from a number to a factor captures the effect of each level. Numbers can sometimes be interpreted as one coefficient instead of multiple coefficients.

bike <- read\_csv("bike\_cleaned.csv")

## Parsed with column specification:  
## cols(  
## instant = col\_double(),  
## dteday = col\_character(),  
## season = col\_character(),  
## mnth = col\_character(),  
## hr = col\_double(),  
## holiday = col\_character(),  
## weekday = col\_character(),  
## workingday = col\_character(),  
## weathersit = col\_character(),  
## temp = col\_double(),  
## atemp = col\_double(),  
## hum = col\_double(),  
## windspeed = col\_double(),  
## casual = col\_double(),  
## registered = col\_double(),  
## count = col\_double()  
## )

str(bike)

## tibble [17,379 x 16] (S3: spec\_tbl\_df/tbl\_df/tbl/data.frame)  
## $ instant : num [1:17379] 1 2 3 4 5 6 7 8 9 10 ...  
## $ dteday : chr [1:17379] "1/1/2011" "1/1/2011" "1/1/2011" "1/1/2011" ...  
## $ season : chr [1:17379] "Winter" "Winter" "Winter" "Winter" ...  
## $ mnth : chr [1:17379] "Jan" "Jan" "Jan" "Jan" ...  
## $ hr : num [1:17379] 0 1 2 3 4 5 6 7 8 9 ...  
## $ holiday : chr [1:17379] "NotHoliday" "NotHoliday" "NotHoliday" "NotHoliday" ...  
## $ weekday : chr [1:17379] "Saturday" "Saturday" "Saturday" "Saturday" ...  
## $ workingday: chr [1:17379] "NotWorkingDay" "NotWorkingDay" "NotWorkingDay" "NotWorkingDay" ...  
## $ weathersit: chr [1:17379] "NoPrecip" "NoPrecip" "NoPrecip" "NoPrecip" ...  
## $ temp : num [1:17379] 0.24 0.22 0.22 0.24 0.24 0.24 0.22 0.2 0.24 0.32 ...  
## $ atemp : num [1:17379] 0.288 0.273 0.273 0.288 0.288 ...  
## $ hum : num [1:17379] 0.81 0.8 0.8 0.75 0.75 0.75 0.8 0.86 0.75 0.76 ...  
## $ windspeed : num [1:17379] 0 0 0 0 0 0.0896 0 0 0 0 ...  
## $ casual : num [1:17379] 3 8 5 3 0 0 2 1 1 8 ...  
## $ registered: num [1:17379] 13 32 27 10 1 1 0 2 7 6 ...  
## $ count : num [1:17379] 16 40 32 13 1 1 2 3 8 14 ...  
## - attr(\*, "spec")=  
## .. cols(  
## .. instant = col\_double(),  
## .. dteday = col\_character(),  
## .. season = col\_character(),  
## .. mnth = col\_character(),  
## .. hr = col\_double(),  
## .. holiday = col\_character(),  
## .. weekday = col\_character(),  
## .. workingday = col\_character(),  
## .. weathersit = col\_character(),  
## .. temp = col\_double(),  
## .. atemp = col\_double(),  
## .. hum = col\_double(),  
## .. windspeed = col\_double(),  
## .. casual = col\_double(),  
## .. registered = col\_double(),  
## .. count = col\_double()  
## .. )

bike = bike %>% mutate(dteday = mdy(dteday))

bike = bike %>% mutate\_if(is.character,as.factor)

bike$hr<-as.factor(bike$hr)  
str(bike)

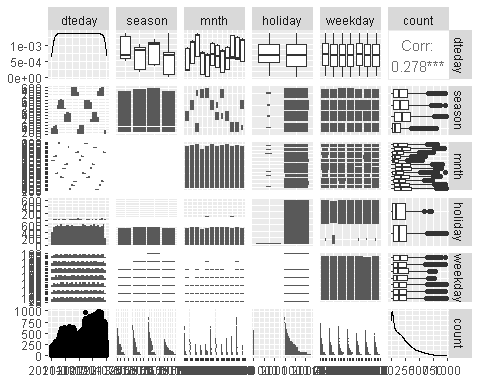
## tibble [17,379 x 16] (S3: spec\_tbl\_df/tbl\_df/tbl/data.frame)  
## $ instant : num [1:17379] 1 2 3 4 5 6 7 8 9 10 ...  
## $ dteday : Date[1:17379], format: "2011-01-01" "2011-01-01" ...  
## $ season : Factor w/ 4 levels "Fall","Spring",..: 4 4 4 4 4 4 4 4 4 4 ...  
## $ mnth : Factor w/ 12 levels "Apr","Aug","Dec",..: 5 5 5 5 5 5 5 5 5 5 ...  
## $ hr : Factor w/ 24 levels "0","1","2","3",..: 1 2 3 4 5 6 7 8 9 10 ...  
## $ holiday : Factor w/ 2 levels "Holiday","NotHoliday": 2 2 2 2 2 2 2 2 2 2 ...  
## $ weekday : Factor w/ 7 levels "Friday","Monday",..: 3 3 3 3 3 3 3 3 3 3 ...  
## $ workingday: Factor w/ 2 levels "NotWorkingDay",..: 1 1 1 1 1 1 1 1 1 1 ...  
## $ weathersit: Factor w/ 4 levels "HeavyPrecip",..: 4 4 4 4 4 3 4 4 4 4 ...  
## $ temp : num [1:17379] 0.24 0.22 0.22 0.24 0.24 0.24 0.22 0.2 0.24 0.32 ...  
## $ atemp : num [1:17379] 0.288 0.273 0.273 0.288 0.288 ...  
## $ hum : num [1:17379] 0.81 0.8 0.8 0.75 0.75 0.75 0.8 0.86 0.75 0.76 ...  
## $ windspeed : num [1:17379] 0 0 0 0 0 0.0896 0 0 0 0 ...  
## $ casual : num [1:17379] 3 8 5 3 0 0 2 1 1 8 ...  
## $ registered: num [1:17379] 13 32 27 10 1 1 0 2 7 6 ...  
## $ count : num [1:17379] 16 40 32 13 1 1 2 3 8 14 ...  
## - attr(\*, "spec")=  
## .. cols(  
## .. instant = col\_double(),  
## .. dteday = col\_character(),  
## .. season = col\_character(),  
## .. mnth = col\_character(),  
## .. hr = col\_double(),  
## .. holiday = col\_character(),  
## .. weekday = col\_character(),  
## .. workingday = col\_character(),  
## .. weathersit = col\_character(),  
## .. temp = col\_double(),  
## .. atemp = col\_double(),  
## .. hum = col\_double(),  
## .. windspeed = col\_double(),  
## .. casual = col\_double(),  
## .. registered = col\_double(),  
## .. count = col\_double()  
## .. )

#Task 2: Which of the quantitative variables appears to be best correlated with “count” (ignore the “registered” and “casual” variable as the sum of these two variables equals “count”)?

ANSWER: temp Note: I had to remove HR because it contained more than 15 levels.

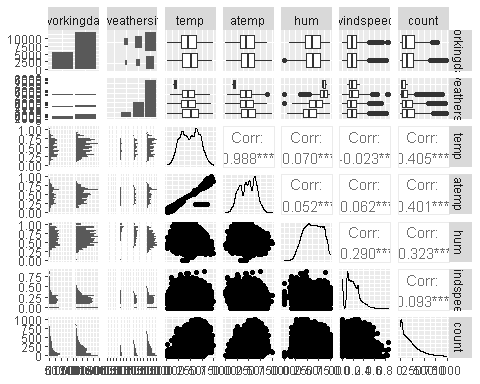
ggpairs(bike, columns = c("dteday", "season", "mnth", "holiday", "weekday", "count"))

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



ggpairs(bike, columns = c("workingday", "weathersit", "temp", "atemp", "hum", "windspeed", "count"))

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



#Task 3: We cannot use correlation to assess the relationship between a categorical predictor variable and our response variable. A good option is to visualize the relationship between the categorical and response variables via a boxplot (or similar visualization). For example, the boxplot for the “hr” variable is shown below (the categorical variable should be on the x-axis): ggplot(bike,aes(x=hr,y=count)) + geom\_boxplot() + theme\_bw()

From this plot, it is fairly obvious that “hr” affects “count”.

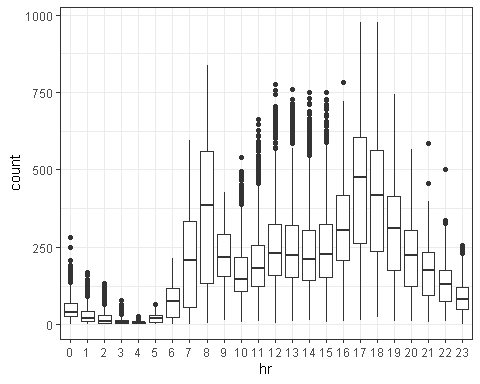
Repeat this boxplot-based analysis for each of the categorical variables. Which variables appear to affect “count”?

ANSWER: HR, Season, Month, Holiday, Workingday, weathersit

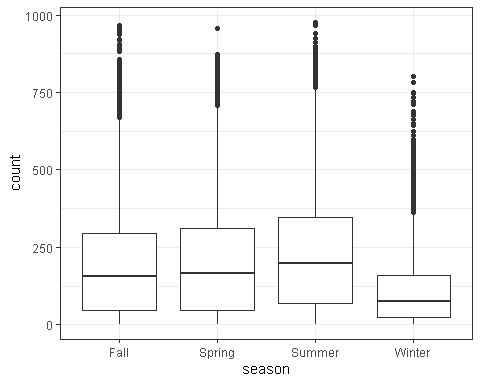
Provide a brief explanation as to why you believe that each variable does or does not affect “count” (use your intuition to help you answer this question).

ANSWER: HR - Does affect count: bike rentals are at its peak during commute hours and limited in the middle of the night. Season - Does affect count: less rentals in the winter Month - Does affect count: less rentals in winter months Holiday - Does affect count: more rentals occur during working days Weekday - Does NOT affect count: rentals are pretty consistent each day of the week Workingday - Does affect count: there are more rentals on work days than weekends weathersit - Does affect count: less rentals when it rains

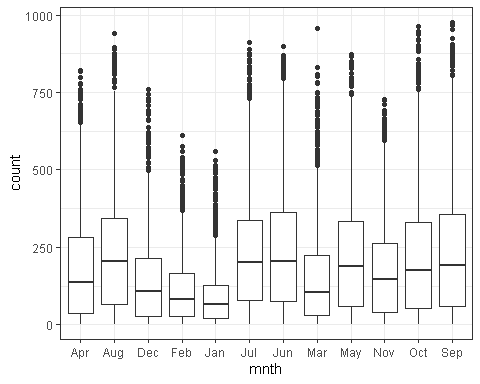
ggplot(bike,aes(x=hr,y=count)) + geom\_boxplot() + theme\_bw()



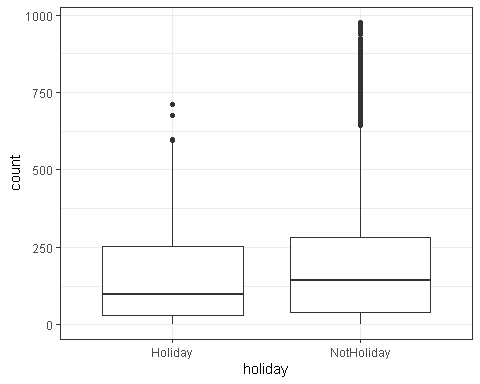
ggplot(bike,aes(x=season,y=count)) + geom\_boxplot() + theme\_bw()



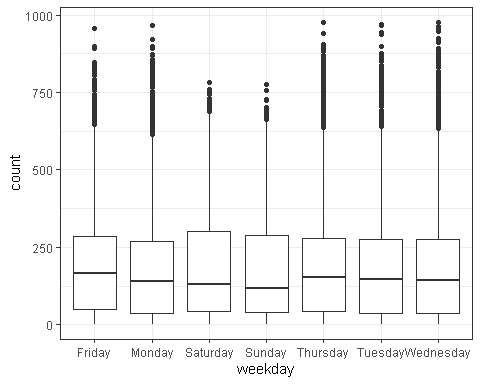
ggplot(bike,aes(x=mnth,y=count)) + geom\_boxplot() + theme\_bw()



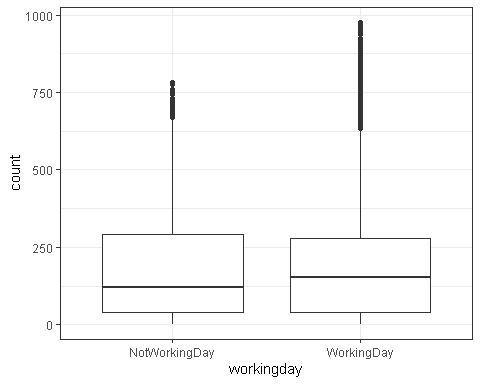
ggplot(bike,aes(x=holiday,y=count)) + geom\_boxplot() + theme\_bw()



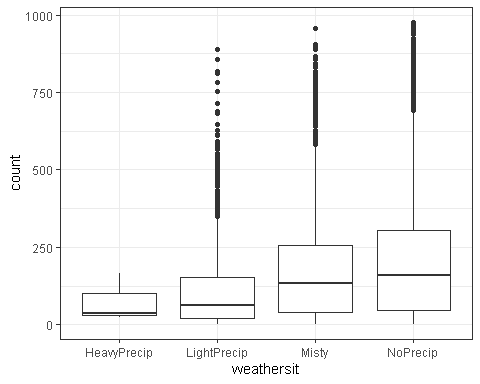
ggplot(bike,aes(x=weekday,y=count)) + geom\_boxplot() + theme\_bw()



ggplot(bike,aes(x=workingday,y=count)) + geom\_boxplot() + theme\_bw()



ggplot(bike,aes(x=weathersit,y=count)) + geom\_boxplot() + theme\_bw()



#Task 4: As a baseline, choose the “best” variable from the correlation and visualization analysis above and build a model with that variable as the single predictor of “count”. Comment on the quality of the model.

ANSWER: R Squared is 0.5008 which is decent. The p-value for each slope coefficient is significant indicating they hr is a significant predictor of count at every hour. It makes sense that hour would impact rental count.

bike\_recipe = recipe(count ~ hr, bike)

lm\_model =   
 linear\_reg() %>%   
 set\_engine("lm")

lm\_wflow =  
 workflow() %>%  
 add\_model(lm\_model) %>%  
 add\_recipe(bike\_recipe)

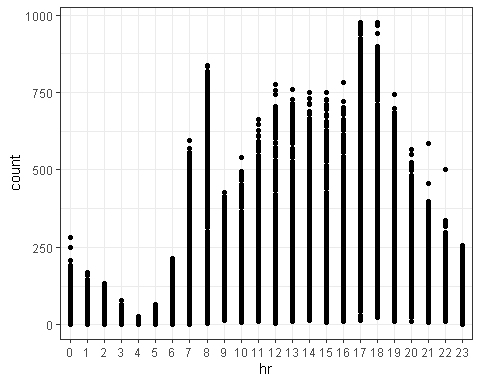
lm\_fit = fit(lm\_wflow, bike)

summary(lm\_fit$fit$fit$fit)

##   
## Call:  
## stats::lm(formula = ..y ~ ., data = data)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -446.45 -60.99 -6.01 50.10 551.49   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 53.898 4.756 11.332 < 2e-16 \*\*\*  
## hr1 -20.522 6.731 -3.049 0.002300 \*\*   
## hr2 -31.028 6.752 -4.595 4.35e-06 \*\*\*  
## hr3 -42.171 6.796 -6.205 5.58e-10 \*\*\*  
## hr4 -47.545 6.796 -6.996 2.73e-12 \*\*\*  
## hr5 -34.008 6.747 -5.040 4.70e-07 \*\*\*  
## hr6 22.146 6.729 3.291 0.000999 \*\*\*  
## hr7 158.167 6.724 23.523 < 2e-16 \*\*\*  
## hr8 305.113 6.724 45.377 < 2e-16 \*\*\*  
## hr9 165.411 6.724 24.600 < 2e-16 \*\*\*  
## hr10 119.770 6.724 17.812 < 2e-16 \*\*\*  
## hr11 154.245 6.724 22.939 < 2e-16 \*\*\*  
## hr12 199.418 6.722 29.668 < 2e-16 \*\*\*  
## hr13 199.763 6.719 29.729 < 2e-16 \*\*\*  
## hr14 187.051 6.719 27.838 < 2e-16 \*\*\*  
## hr15 197.335 6.719 29.368 < 2e-16 \*\*\*  
## hr16 258.085 6.717 38.422 < 2e-16 \*\*\*  
## hr17 407.554 6.717 60.674 < 2e-16 \*\*\*  
## hr18 371.613 6.722 55.286 < 2e-16 \*\*\*  
## hr19 257.625 6.722 38.327 < 2e-16 \*\*\*  
## hr20 172.132 6.722 25.608 < 2e-16 \*\*\*  
## hr21 118.416 6.722 17.617 < 2e-16 \*\*\*  
## hr22 77.437 6.722 11.520 < 2e-16 \*\*\*  
## hr23 33.933 6.722 5.048 4.50e-07 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 128.2 on 17355 degrees of freedom  
## Multiple R-squared: 0.5015, Adjusted R-squared: 0.5008   
## F-statistic: 759.1 on 23 and 17355 DF, p-value: < 2.2e-16

ggplot(bike, aes(x=hr, y=count)) + geom\_point() + geom\_smooth(method = lm, se = FALSE) + theme\_bw()

## `geom\_smooth()` using formula 'y ~ x'



#Task 5:

Create a ridge regression model to predict the “count” variable. You should exclude the “instant”, “dteday”, “registered”, and “casual” variables (i.e., they should not be predictors). You may apply any appropriate preprocessing steps. HINT: You can use “step\_rm” in the recipe to exclude variables from analyis. Select an appropriate value for lambda.

Provide a brief commentary on the resulting model.

ANSWER: I selected a lambda of 11. The y-intercept is 189.4630. In looking at hrs, the slope is negative from 10pm to 6am (meaning low bike rentals during these hours). The slope is greatest (meaning most bike rentals occur) at 5pm followed by 6pm and 8am which are peak hours for people renting bikes to get to and from work.

bike\_recipe2 = recipe(count ~ ., bike) %>%  
 step\_rm(instant,dteday,registered,casual) %>%  
 step\_dummy(all\_nominal()) %>%  
 step\_center(all\_predictors()) %>%  
 step\_scale(all\_predictors())

ridge\_model =  
 linear\_reg(mixture = 0) %>% #mixture = 0 sets up Ridge  
 set\_engine("glmnet")

ridge\_wflow =  
 workflow() %>%  
 add\_model(ridge\_model) %>%  
 add\_recipe(bike\_recipe2)

ridge\_fit = fit(ridge\_wflow, bike)

ridge\_fit %>%  
 pull\_workflow\_fit() %>%  
 pluck("fit") #select best lambda value

##   
## Call: glmnet::glmnet(x = maybe\_matrix(x), y = y, family = "gaussian", alpha = ~0)   
##   
## Df %Dev Lambda  
## 1 52 0.00 73420  
## 2 52 0.61 66900  
## 3 52 0.67 60950  
## 4 52 0.74 55540  
## 5 52 0.81 50600  
## 6 52 0.88 46110  
## 7 52 0.97 42010  
## 8 52 1.06 38280  
## 9 52 1.16 34880  
## 10 52 1.27 31780  
## 11 52 1.39 28960  
## 12 52 1.53 26390  
## 13 52 1.67 24040  
## 14 52 1.83 21910  
## 15 52 2.00 19960  
## 16 52 2.19 18190  
## 17 52 2.40 16570  
## 18 52 2.62 15100  
## 19 52 2.86 13760  
## 20 52 3.13 12540  
## 21 52 3.41 11420  
## 22 52 3.72 10410  
## 23 52 4.06 9482  
## 24 52 4.43 8640  
## 25 52 4.83 7872  
## 26 52 5.26 7173  
## 27 52 5.72 6536  
## 28 52 6.22 5955  
## 29 52 6.76 5426  
## 30 52 7.34 4944  
## 31 52 7.96 4505  
## 32 52 8.62 4105  
## 33 52 9.33 3740  
## 34 52 10.09 3408  
## 35 52 10.90 3105  
## 36 52 11.76 2829  
## 37 52 12.67 2578  
## 38 52 13.63 2349  
## 39 52 14.65 2140  
## 40 52 15.72 1950  
## 41 52 16.83 1777  
## 42 52 18.01 1619  
## 43 52 19.23 1475  
## 44 52 20.49 1344  
## 45 52 21.81 1225  
## 46 52 23.16 1116  
## 47 52 24.56 1017  
## 48 52 25.98 926  
## 49 52 27.44 844  
## 50 52 28.93 769  
## 51 52 30.43 701  
## 52 52 31.95 639  
## 53 52 33.48 582  
## 54 52 35.01 530  
## 55 52 36.53 483  
## 56 52 38.04 440  
## 57 52 39.54 401  
## 58 52 41.01 365  
## 59 52 42.44 333  
## 60 52 43.84 303  
## 61 52 45.20 276  
## 62 52 46.51 252  
## 63 52 47.77 230  
## 64 52 48.96 209  
## 65 52 50.10 190  
## 66 52 51.18 174  
## 67 52 52.19 158  
## 68 52 53.14 144  
## 69 52 54.02 131  
## 70 52 54.83 120  
## 71 52 55.59 109  
## 72 52 56.28 99  
## 73 52 56.91 91  
## 74 52 57.49 82  
## 75 52 58.01 75  
## 76 52 58.48 68  
## 77 52 58.91 62  
## 78 52 59.30 57  
## 79 52 59.64 52  
## 80 52 59.96 47  
## 81 52 60.24 43  
## 82 52 60.49 39  
## 83 52 60.72 36  
## 84 52 60.93 33  
## 85 52 61.11 30  
## 86 52 61.28 27  
## 87 52 61.44 25  
## 88 52 61.58 22  
## 89 52 61.71 20  
## 90 52 61.83 19  
## 91 52 61.95 17  
## 92 52 62.05 15  
## 93 52 62.14 14  
## 94 52 62.23 13  
## 95 52 62.32 12  
## 96 52 62.40 11  
## 97 52 62.47 10  
## 98 52 62.54 9  
## 99 52 62.60 8  
## 100 52 62.66 7

ridge\_fit %>%  
 pull\_workflow\_fit() %>%  
 pluck("fit") %>%   
 coef(s = 11) #show the coefficients for our selected lambda value

## 53 x 1 sparse Matrix of class "dgCMatrix"  
## 1  
## (Intercept) 189.46308763  
## temp 27.54095517  
## atemp 25.53171982  
## hum -24.24502823  
## windspeed -3.52392172  
## season\_Spring -4.45265692  
## season\_Summer -9.64806956  
## season\_Winter -19.29716985  
## mnth\_Aug -0.61711593  
## mnth\_Dec 1.57206781  
## mnth\_Feb -0.10308267  
## mnth\_Jan -0.06336942  
## mnth\_Jul -7.32763843  
## mnth\_Jun -2.54368720  
## mnth\_Mar 1.93273440  
## mnth\_May 2.78228818  
## mnth\_Nov 1.90855013  
## mnth\_Oct 7.54549613  
## mnth\_Sep 8.28460509  
## hr\_X1 -16.58100247  
## hr\_X2 -17.94610339  
## hr\_X3 -19.54444255  
## hr\_X4 -19.86455979  
## hr\_X5 -17.07505749  
## hr\_X6 -6.13016681  
## hr\_X7 19.18698009  
## hr\_X8 45.39527560  
## hr\_X9 16.92341929  
## hr\_X10 6.03983748  
## hr\_X11 10.31062095  
## hr\_X12 17.29697663  
## hr\_X13 16.06458520  
## hr\_X14 12.86933945  
## hr\_X15 14.60580127  
## hr\_X16 26.46613470  
## hr\_X17 55.83900369  
## hr\_X18 50.02101590  
## hr\_X19 29.92000400  
## hr\_X20 15.21236551  
## hr\_X21 6.20806927  
## hr\_X22 -0.51767734  
## hr\_X23 -7.65758060  
## holiday\_NotHoliday 3.54475733  
## weekday\_Monday -2.13761175  
## weekday\_Saturday 1.63623802  
## weekday\_Sunday -3.10703356  
## weekday\_Thursday -1.17540435  
## weekday\_Tuesday -1.63833346  
## weekday\_Wednesday -0.68938269  
## workingday\_WorkingDay 2.40030783  
## weathersit\_LightPrecip -12.15397947  
## weathersit\_Misty 2.38113804  
## weathersit\_NoPrecip 4.81632254

#Task 6: Create a lasso regression model to predict the “count” variable. You should exclude the “instant”, “dteday”, “registered”, and “casual” variables (i.e., they should not be predictors). You may apply any appropriate preprocessing steps. Select an appropriate value for lambda.

Provide a brief commentary on the resulting model.

ANSWER: I selected a lambda of 0.190. The y-intercept is 189.4630 (this is the same y-intercept we got using the Ridge model. In looking at hrs, the slope is negative from 1am to 5am (meaning low bike rentals during these hours). Like Ridge, the slope is greatest (meaning most bike rentals occur) at 5pm followed by 6pm and 8am which are peak hours for people renting bikes to get to and from work.

What are the implications of the model results from the ridge and lasso methods?

ANSWER: While the y-intercept remained the same when running the Ridge and Lasso models, the slopes differed. The slopes using the lasso model where small in the night hours and bigger in the peak day hours. For instance the slope at 5am using the Ridge model was -17.0750 and using the Lasso model was -6.5397. The slope at 5pm using the Ridge model was 55.8390 and using the Lasso model was 70.7751.

lasso\_model =   
 linear\_reg(mixture = 1) %>% #mixture = 1 sets up Lasso  
 set\_engine("glmnet")

lasso\_wflow =   
 workflow() %>%   
 add\_model(lasso\_model) %>%   
 add\_recipe(bike\_recipe2)

lasso\_fit = fit(lasso\_wflow, bike)

lasso\_fit %>%  
 pull\_workflow\_fit() %>%  
 pluck("fit") # select lambda

##   
## Call: glmnet::glmnet(x = maybe\_matrix(x), y = y, family = "gaussian", alpha = ~1)   
##   
## Df %Dev Lambda  
## 1 0 0.00 73.420  
## 2 1 2.78 66.900  
## 3 1 5.09 60.950  
## 4 3 7.60 55.540  
## 5 3 11.69 50.600  
## 6 4 15.44 46.110  
## 7 4 19.18 42.010  
## 8 6 22.56 38.280  
## 9 6 26.23 34.880  
## 10 6 29.28 31.780  
## 11 8 32.06 28.960  
## 12 11 34.97 26.390  
## 13 12 38.11 24.040  
## 14 12 40.86 21.910  
## 15 14 43.28 19.960  
## 16 14 45.50 18.190  
## 17 15 47.37 16.570  
## 18 15 49.03 15.100  
## 19 16 50.55 13.760  
## 20 16 51.81 12.540  
## 21 18 52.98 11.420  
## 22 19 54.01 10.410  
## 23 21 54.90 9.482  
## 24 24 55.78 8.640  
## 25 25 56.58 7.872  
## 26 26 57.29 7.173  
## 27 27 57.91 6.536  
## 28 27 58.47 5.955  
## 29 28 58.95 5.426  
## 30 28 59.38 4.944  
## 31 29 59.74 4.505  
## 32 31 60.09 4.105  
## 33 32 60.41 3.740  
## 34 32 60.69 3.408  
## 35 32 60.92 3.105  
## 36 33 61.11 2.829  
## 37 36 61.30 2.578  
## 38 37 61.60 2.349  
## 39 36 61.82 2.140  
## 40 36 61.98 1.950  
## 41 38 62.13 1.777  
## 42 39 62.25 1.619  
## 43 40 62.36 1.475  
## 44 41 62.46 1.344  
## 45 42 62.58 1.225  
## 46 42 62.69 1.116  
## 47 42 62.77 1.017  
## 48 41 62.84 0.926  
## 49 42 62.89 0.844  
## 50 42 62.92 0.769  
## 51 42 62.96 0.701  
## 52 42 62.98 0.639  
## 53 42 63.01 0.582  
## 54 42 63.04 0.530  
## 55 42 63.05 0.483  
## 56 43 63.07 0.440  
## 57 44 63.09 0.401  
## 58 45 63.11 0.365  
## 59 45 63.13 0.333  
## 60 45 63.14 0.303  
## 61 46 63.15 0.276  
## 62 49 63.16 0.252  
## 63 49 63.17 0.230  
## 64 49 63.18 0.209  
## 65 49 63.19 0.190  
## 66 49 63.19 0.174  
## 67 49 63.20 0.158  
## 68 49 63.20 0.144  
## 69 49 63.21 0.131  
## 70 48 63.21 0.120  
## 71 48 63.21 0.109  
## 72 48 63.21 0.099  
## 73 48 63.22 0.091  
## 74 49 63.22 0.082  
## 75 49 63.22 0.075  
## 76 49 63.22 0.068  
## 77 49 63.22 0.062  
## 78 49 63.22 0.057  
## 79 50 63.22 0.052  
## 80 50 63.22 0.047  
## 81 50 63.22 0.043

lasso\_fit %>%  
 pull\_workflow\_fit() %>%  
 pluck("fit") %>%   
 coef(s = 0.190) #show the coefficients for our selected lambda value

## 53 x 1 sparse Matrix of class "dgCMatrix"  
## 1  
## (Intercept) 189.4630876  
## temp 34.6874495  
## atemp 19.4624332  
## hum -22.5452659  
## windspeed -4.9138257  
## season\_Spring -8.9051469  
## season\_Summer -14.7656401  
## season\_Winter -24.3777727  
## mnth\_Aug -0.7377979  
## mnth\_Dec .   
## mnth\_Feb 0.5750304  
## mnth\_Jan 0.6150240  
## mnth\_Jul -7.7224664  
## mnth\_Jun -2.7166764  
## mnth\_Mar 2.4656693  
## mnth\_May 2.4131520  
## mnth\_Nov -0.4764556  
## mnth\_Oct 4.8117561  
## mnth\_Sep 7.5746275  
## hr\_X1 -5.9113423  
## hr\_X2 -7.4415864  
## hr\_X3 -9.2932502  
## hr\_X4 -9.6357683  
## hr\_X5 -6.5397377  
## hr\_X6 4.7587641  
## hr\_X7 31.6114681  
## hr\_X8 59.3954949  
## hr\_X9 29.3419608  
## hr\_X10 17.8852442  
## hr\_X11 22.4601065  
## hr\_X12 29.9273729  
## hr\_X13 28.6673172  
## hr\_X14 25.3210933  
## hr\_X15 27.1656410  
## hr\_X16 39.7142127  
## hr\_X17 70.7751228  
## hr\_X18 64.5633985  
## hr\_X19 43.2190999  
## hr\_X20 27.6074461  
## hr\_X21 18.0075124  
## hr\_X22 10.8477020  
## hr\_X23 3.2628654  
## holiday\_NotHoliday 4.3237007  
## weekday\_Monday -1.8166561  
## weekday\_Saturday 0.1604665  
## weekday\_Sunday -4.6396519  
## weekday\_Thursday -0.7341056  
## weekday\_Tuesday -1.3148368  
## weekday\_Wednesday -0.2903128  
## workingday\_WorkingDay .   
## weathersit\_LightPrecip -14.6518373  
## weathersit\_Misty .   
## weathersit\_NoPrecip 2.6651757

```