#install.packages("car")  
#install.packages("ggcorrplot")  
#install.packages("glmnet")

library(tidyverse)

## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.0 ──

## ✓ ggplot2 3.3.2 ✓ purrr 0.3.4  
## ✓ tibble 3.0.4 ✓ dplyr 1.0.2  
## ✓ tidyr 1.1.2 ✓ stringr 1.4.0  
## ✓ readr 1.4.0 ✓ 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 ──

## ✓ broom 0.7.3 ✓ recipes 0.1.15  
## ✓ dials 0.0.9 ✓ rsample 0.0.8   
## ✓ infer 0.5.4 ✓ tune 0.1.2   
## ✓ modeldata 0.1.0 ✓ workflows 0.2.1   
## ✓ parsnip 0.1.5 ✓ 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(glmnet)

## Loading required package: Matrix

##   
## Attaching package: 'Matrix'

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

## Loaded glmnet 4.1

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

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

##   
## ── 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()  
## )

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

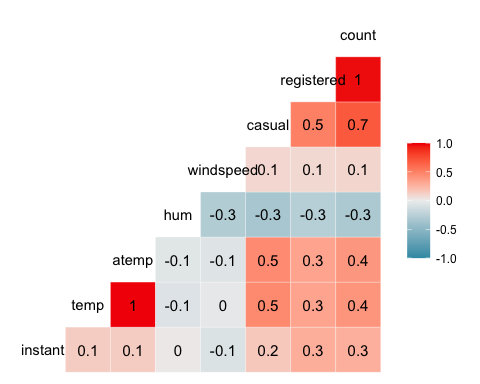
bike = bike %>% mutate(season = factor(season))  
bike = bike %>% mutate(mnth = factor(mnth))  
bike = bike %>% mutate(holiday = factor(holiday))  
bike = bike %>% mutate(weekday = factor(weekday))  
bike = bike %>% mutate(workingday = factor(workingday))  
bike = bike %>% mutate(weathersit = factor(weathersit))  
bike = bike %>% mutate(hr = factor(hr))

Why do we convert the “hr” variable into factor? Why not just leave as numbers? “hr” should be converted to a factor because it is categorical data, thus it does not need to be left as a number.

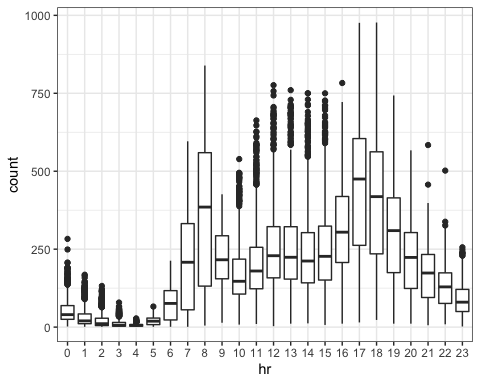
ggcorr(bike, label= "TRUE", Label\_round = 2)

## Warning in ggcorr(bike, label = "TRUE", Label\_round = 2): data in column(s)  
## 'dteday', 'season', 'mnth', 'hr', 'holiday', 'weekday', 'workingday',  
## 'weathersit' are not numeric and were ignored

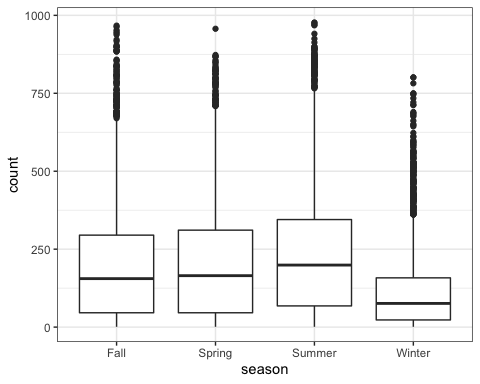
## Warning: Ignoring unknown parameters: Label\_round

 Which of the quantitative variables appears to be best correlated with “count”? “temp” is most correlated with “count”.

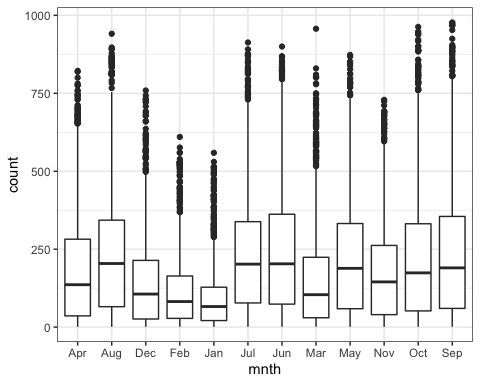
ggplot(bike,aes(x=hr,y=count)) + geom\_boxplot() + theme\_bw()

 “count” is affected by “hr”. This makes logical sense, because people are the count goes up in the middle of the day and drops in the middle of the night. This shows hours of operations and the busiest times.

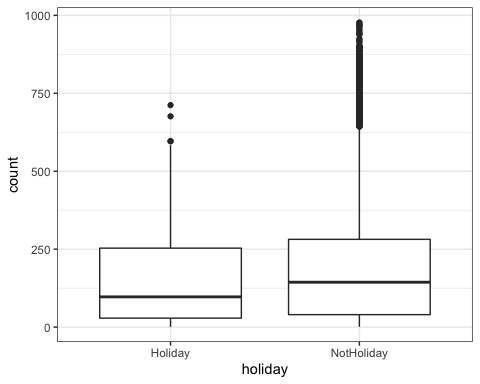
ggplot(bike,aes(x=season,y=count)) + geom\_boxplot() + theme\_bw()

 “count” is affected by season, this also makes logical sense, bcause people are using this service less in the winter, and more in the summer, spring, and fall.

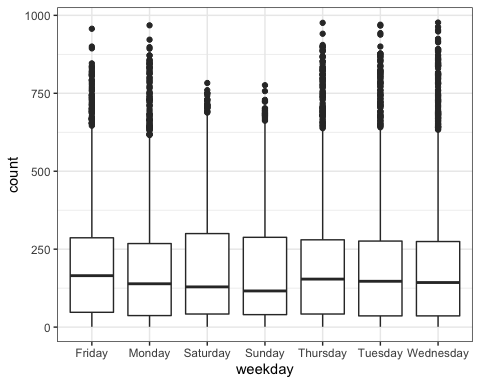
ggplot(bike,aes(x=mnth,y=count)) + geom\_boxplot() + theme\_bw()

 “count” is affected by “mnth”. This makes sense, because the warmer months have a higher count and the cold winter months have a lower count.

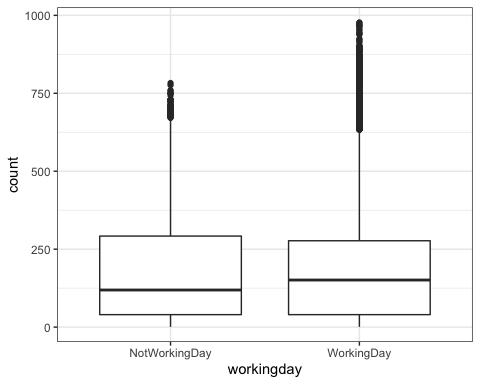
ggplot(bike,aes(x=holiday,y=count)) + geom\_boxplot() + theme\_bw()

 “count” is affected by “holiday”. The holiday days show a lower count than NotHoliday days. This could be because people have other obligations on Holidays.

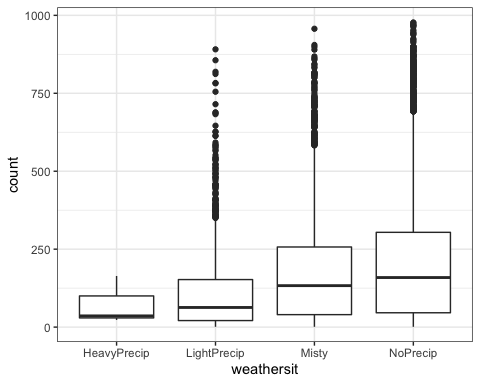
ggplot(bike,aes(x=weekday,y=count)) + geom\_boxplot() + theme\_bw()

 “count” is affected by “weekday”. This is not by a large amount, but you can see a change from the weekdays and the weekend days.

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

 “count” is affected by “NotWorkingDay” and “WorkingDay”, but by a small amount. This is litte change in count between the two.

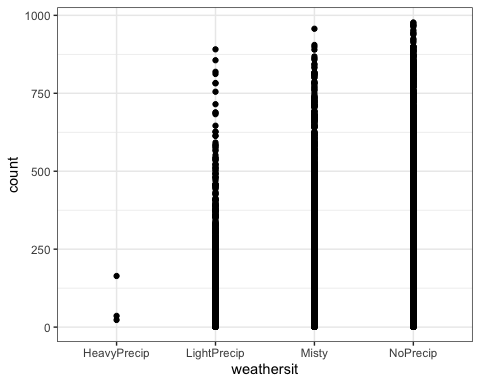
ggplot(bike,aes(x=weathersit,y=count)) + geom\_boxplot() + theme\_bw()

 “count” is largely affected by “weathersit”. You can see that no one is using this service during heavy percip, but the count is the highest when there is No percip.

bike\_recipe = recipe(count ~ weathersit, bike)  
  
lm\_model = #give the model type a name   
 linear\_reg() %>% #specify that we are doing linear regression  
 set\_engine("lm") #specify the specify type of linear tool we want to use   
  
lm\_wflow =   
 workflow() %>%   
 add\_model(lm\_model) %>%   
 add\_recipe(bike\_recipe)  
  
lm\_fit = fit(lm\_wflow, bike)

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

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

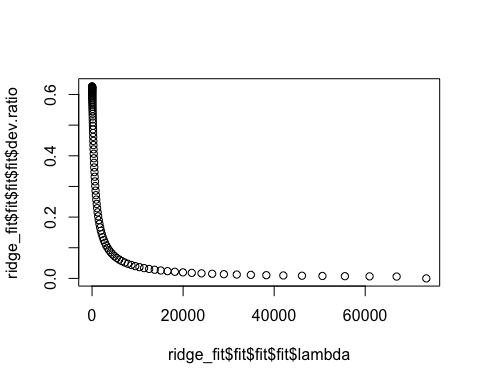
 This model shows that when there is heavy precipitation the count is very low. The count is high when there is light to no precipitation.

bike\_recipe = recipe(count ~. , bike) %>%  
 step\_rm(instant,dteday,registered,casual) %>%  
 step\_other(weathersit, threshold = 0.01) %>%   
 step\_dummy(all\_nominal()) %>%   
 step\_center(all\_predictors()) %>%   
 step\_scale(all\_predictors())   
   
ridge\_model =   
 linear\_reg(mixture = 0 ) %>%   
 set\_engine("glmnet")   
  
ridge\_wflow =   
 workflow() %>%   
 add\_model(ridge\_model) %>%   
 add\_recipe(bike\_recipe)  
  
ridge\_fit = fit(ridge\_wflow, bike)

ridge\_fit

## ══ Workflow [trained] ══════════════════════════════════════════════════════════  
## Preprocessor: Recipe  
## Model: linear\_reg()  
##   
## ── Preprocessor ────────────────────────────────────────────────────────────────  
## 5 Recipe Steps  
##   
## ● step\_rm()  
## ● step\_other()  
## ● step\_dummy()  
## ● step\_center()  
## ● step\_scale()  
##   
## ── Model ───────────────────────────────────────────────────────────────────────  
##   
## Call: glmnet::glmnet(x = maybe\_matrix(x), y = y, family = "gaussian", alpha = ~0)   
##   
## Df %Dev Lambda  
## 1 52 0.00 73420  
## 2 52 0.60 66900  
## 3 52 0.66 60950  
## 4 52 0.73 55540  
## 5 52 0.79 50600  
## 6 52 0.87 46110  
## 7 52 0.95 42010  
## 8 52 1.05 38280  
## 9 52 1.15 34880  
## 10 52 1.26 31780  
## 11 52 1.38 28960  
## 12 52 1.51 26390  
## 13 52 1.65 24040  
## 14 52 1.80 21910  
## 15 52 1.97 19960  
## 16 52 2.16 18190  
## 17 52 2.36 16570  
## 18 52 2.58 15100  
## 19 52 2.82 13760  
## 20 52 3.08 12540  
## 21 52 3.37 11420  
## 22 52 3.67 10410  
## 23 52 4.01 9482  
## 24 52 4.37 8640  
## 25 52 4.76 7872  
## 26 52 5.18 7173  
## 27 52 5.64 6536  
## 28 52 6.13 5955  
## 29 52 6.67 5426  
## 30 52 7.24 4944  
## 31 52 7.85 4505  
## 32 52 8.51 4105  
## 33 52 9.21 3740  
## 34 52 9.96 3408  
## 35 52 10.76 3105  
## 36 52 11.61 2829  
## 37 52 12.51 2578  
## 38 52 13.46 2349  
## 39 52 14.47 2140  
## 40 52 15.52 1950  
## 41 52 16.63 1777  
## 42 52 17.79 1619  
## 43 52 19.00 1475  
## 44 52 20.26 1344  
## 45 52 21.56 1225  
## 46 52 22.90 1116  
##   
## ...  
## and 54 more lines.

plot(ridge\_fit$fit$fit$fit$lambda,ridge\_fit$fit$fit$fit$dev.ratio)



The dev.ratio when lambda is closest to zero is 0.6. This makes sense, because the correlation model showed a correlation of about 0.6.

bike\_recipe = recipe(count ~., bike) %>%   
 step\_rm(instant,dteday,registered, casual)%>%  
step\_other(weathersit, threshold = 0.01) %>%   
 step\_dummy(all\_nominal()) %>%   
 step\_center(all\_predictors()) %>%   
 step\_scale(all\_predictors())   
   
lasso\_model =  
 linear\_reg(mixture = 1) %>%   
 set\_engine("glmnet")   
  
lasso\_wflow =   
 workflow() %>%   
 add\_model(lasso\_model) %>%   
 add\_recipe(bike\_recipe)  
  
lasso\_fit = fit(lasso\_wflow, bike)

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

##   
## 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 17 52.90 11.420  
## 22 18 53.85 10.410  
## 23 20 54.67 9.482  
## 24 23 55.51 8.640  
## 25 24 56.26 7.872  
## 26 25 56.93 7.173  
## 27 25 57.51 6.536  
## 28 26 58.03 5.955  
## 29 26 58.46 5.426  
## 30 27 58.86 4.944  
## 31 27 59.20 4.505  
## 32 30 59.53 4.105  
## 33 32 59.85 3.740  
## 34 33 60.22 3.408  
## 35 35 60.53 3.105  
## 36 35 60.81 2.829  
## 37 38 61.07 2.578  
## 38 38 61.40 2.349  
## 39 37 61.65 2.140  
## 40 38 61.86 1.950  
## 41 39 62.03 1.777  
## 42 39 62.16 1.619  
## 43 40 62.29 1.475  
## 44 41 62.40 1.344  
## 45 43 62.53 1.225  
## 46 42 62.64 1.116  
## 47 42 62.74 1.017  
## 48 41 62.81 0.926  
## 49 41 62.86 0.844  
## 50 42 62.90 0.769  
## 51 42 62.94 0.701  
## 52 43 62.97 0.639  
## 53 42 63.00 0.582  
## 54 42 63.03 0.530  
## 55 42 63.05 0.483  
## 56 43 63.06 0.440  
## 57 44 63.09 0.401  
## 58 45 63.11 0.365  
## 59 45 63.12 0.333  
## 60 46 63.14 0.303  
## 61 48 63.15 0.276  
## 62 50 63.16 0.252  
## 63 50 63.17 0.230  
## 64 50 63.18 0.209  
## 65 50 63.19 0.190  
## 66 50 63.19 0.174  
## 67 50 63.20 0.158  
## 68 50 63.20 0.144  
## 69 50 63.21 0.131  
## 70 49 63.21 0.120  
## 71 49 63.21 0.109  
## 72 49 63.21 0.099  
## 73 49 63.22 0.091  
## 74 49 63.22 0.082  
## 75 50 63.22 0.075  
## 76 50 63.22 0.068  
## 77 50 63.22 0.062  
## 78 50 63.22 0.057  
## 79 51 63.22 0.052  
## 80 51 63.22 0.047  
## 81 51 63.22 0.043

lasso\_fit %>%  
 pull\_workflow\_fit() %>%  
 pluck("fit") %>%   
 coef(s = 3.408)

## 53 x 1 sparse Matrix of class "dgCMatrix"  
## 1  
## (Intercept) 189.4630876  
## temp 13.0324876  
## atemp 35.5572126  
## hum -29.4346322  
## windspeed .   
## season\_Spring .   
## season\_Summer -2.3344295  
## season\_Winter -17.1202698  
## mnth\_Aug .   
## mnth\_Dec .   
## mnth\_Feb .   
## mnth\_Jan .   
## mnth\_Jul -5.4251665  
## mnth\_Jun .   
## mnth\_Mar .   
## mnth\_May .   
## mnth\_Nov .   
## mnth\_Oct 6.4496344  
## mnth\_Sep 5.2567235  
## hr\_X1 -19.0482900  
## hr\_X2 -20.4434172  
## hr\_X3 -21.9998565  
## hr\_X4 -22.3815770  
## hr\_X5 -19.3655887  
## hr\_X6 -7.9413311  
## hr\_X7 12.0300679  
## hr\_X8 39.7481651  
## hr\_X9 9.4690074  
## hr\_X10 .   
## hr\_X11 2.2277527  
## hr\_X12 9.3640338  
## hr\_X13 7.9385266  
## hr\_X14 4.4402822  
## hr\_X15 6.2061050  
## hr\_X16 18.7325140  
## hr\_X17 49.7920686  
## hr\_X18 43.8158173  
## hr\_X19 22.8537149  
## hr\_X20 7.3379706  
## hr\_X21 .   
## hr\_X22 -2.3505814  
## hr\_X23 -9.9443353  
## holiday\_NotHoliday 0.7218011  
## weekday\_Monday .   
## weekday\_Saturday .   
## weekday\_Sunday -0.2118374  
## weekday\_Thursday .   
## weekday\_Tuesday .   
## weekday\_Wednesday .   
## workingday\_WorkingDay .   
## weathersit\_Misty 2.2904338  
## weathersit\_NoPrecip 3.5576668  
## weathersit\_other .

I choose 3.408 for Lambda. It seemed to me that this was a good leveling off point. The model above explains the dataset further.