Problem Set 8

1.

a.

1. Let data be seen by different individuals so that they can correct mistakes that you may have made in your data set. To base conclusions off of a bad set or just a simple mistake can have very severe and lasting consequences.
2. Be as objective as possible when presenting data so that new individuals can make their own assertions on the data rather than borrow what was said by the author of the data.
3. Understand what your data is and present it as optimal as possible. No biases should be presented within the data so that when someone does a random sample, they're conclusions are not preset on the authors inherit biases.
4. If you are going to critique someones data, have some supporting evidence to back your claim. Either supported by something scientific or analyzed by some other professional. To blatanly critique someones presented data is fundementally problematic.
5. Statistics are not the ultimatum when stating a discovery or resolution. To simply present your analysis says nothing to someone that does not understand the statistics. Make sure you are clear about how you present your data and make assertations that are easy to follow.

b.

1. Make sure you take your data to different departments to be examined and tested to ensure your team is not overlooking a simple mistake
2. Be very mindful of where you got the data and ensure all data being analyzed is somewhat relevant to what you are attempting to analyze.
3. Avoid any sort of biases that may be inherit in the team about the direction of the company. Ensure the data has not been tampered with and accurately reflects your companies capabilities.
4. When critiquing the work of an associate, ensure that the critique is supported by some form of evidence that they may have overlooked in the data.
5. When presenting the advertisement data analysis, ensure that you are using language that is easily accessible to individuals who don't have the same level of sophisitication as yourself.

2

```
DayDF <- read.csv("/Users/karlhickel/Desktop/Bike-Sharing-Dataset/day.csv")
```

d

```
DayDF$weekday <- as.factor(DayDF$weekday)
DayDF$workingday <- as.factor(DayDF$workingday)
DayDF$weathersit <- as.factor(DayDF$weathersit)
DayDF$season <- as.factor(DayDF$season)
DayDF$yr <- as.factor(DayDF$yr)
DayDF$mnth <- as.factor(DayDF$mnth)
DayDF$holiday <- as.factor(DayDF$holiday)
```

e.

```
DayDF$tempSq <- DayDF$temp * DayDF$temp
DayDF$atempSq <- DayDF$atemp * DayDF$atemp
DayDF$humSq <- DayDF$hum * DayDF$hum
DayDF$windspeedSq <- DayDF$windspeed * DayDF$windspeed
```

f.

```
myFormula <- as.formula(casual ~ season + yr + mnth + holiday + weekday +
workingday + weathersit + temp + tempSq + atemp + atempSq + hum + humSq +
windspeed + windspeedSq)
```

g.

```
#install.packages("useful")
library(useful)
```

```
## Loading required package: ggplot2
```

```
xVars <- build.x(myFormula,data = DayDF)
yVars <- build.y(myFormula, data = DayDF)
```

h.

So by seperating my independently factored varibales, by doing this it creates more columns but keeps the same number of instances.

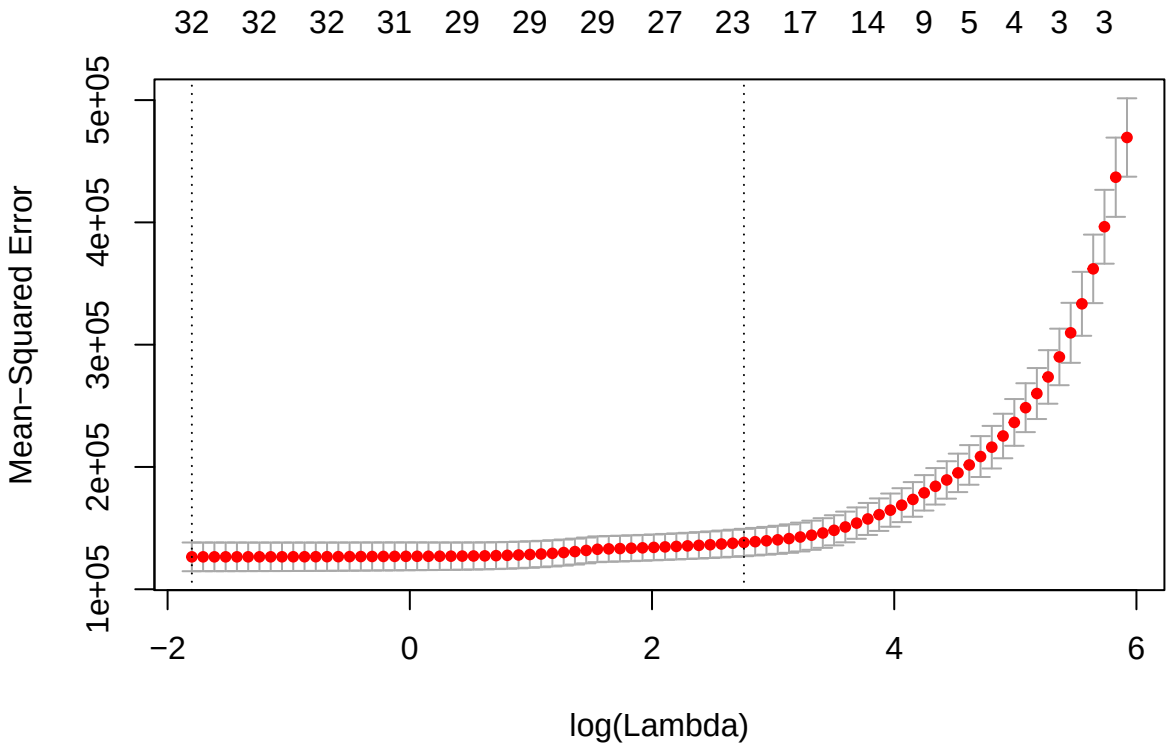
i.

```
#install.packages("glmnet")
set.seed(1861)
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loading required package: foreach
## Loaded glmnet 2.0-16
```

```
LassoMod <- cv.glmnet(x = xVars, y = yVars , alpha =  1)
plot(LassoMod)
```



j.

```
coef(LassoMod, s = "lambda.min")
```

```
## 35 x 1 sparse Matrix of class "dgCMatrix"
##                          1
## (Intercept)  -168.040528
## (Intercept)     .
## season2       210.433862
## season3       119.518818
## season4        50.165732
## yr1           265.695949
## mnth2         -94.736059
## mnth3         138.016527
## mnth4         101.583352
## mnth5         155.246572
## mnth6          52.969314
## mnth7          84.352440
## mnth8          78.392780
## mnth9         187.654502
## mnth10        207.809787
## mnth11          7.652538
```

```
## mnth12        -99.071931
## holiday1     -216.286739
## weekday1        .
## weekday2      -58.329490
## weekday3      -63.347543
## weekday4      -50.429412
## weekday5      113.513326
## weekday6      150.114135
## workingday1  -742.053358
## weathersit2   -91.703583
## weathersit3  -223.792809
## temp         4517.048936
## tempSq      -2503.900731
## atemp         161.678691
## atempSq      -815.782907
## hum           745.950510
## humSq       -1137.502892
## windspeed    -399.096915
## windspeedSq -1791.924014
```

```
coef(LassoMod, s = "lambda.1se")
```

```
## 35 x 1 sparse Matrix of class "dgCMatrix"
##                         1
## (Intercept)   497.587204
## (Intercept)     .
## season2       210.474192
## season3         .
## season4         7.009135
## yr1           256.418875
## mnth2        -110.668501
## mnth3          50.100147
## mnth4          30.881539
## mnth5          29.271152
## mnth6           .
## mnth7           .
## mnth8           .
## mnth9         146.414979
## mnth10        179.551072
## mnth11          .
## mnth12        -70.326872
## holiday1     -120.626696
## weekday1        .
## weekday2       -5.237374
## weekday3       -9.527506
## weekday4        .
## weekday5      101.602454
## weekday6      139.716822
## workingday1  -727.753723
## weathersit2   -71.702483
## weathersit3  -213.676670
## temp         1332.182645
## tempSq          .
## atemp         505.982532
## atempSq         .
```

```
## hum             .
## humSq        -356.181440
## windspeed    -275.577914
## windspeedSq -1301.599435
```

k.

```
coef(LassoMod, s = "lambda.min")
```

```
## 35 x 1 sparse Matrix of class "dgCMatrix"
##                         1
## (Intercept)  -168.040528
## (Intercept)     .
## season2       210.433862
## season3       119.518818
## season4        50.165732
## yr1           265.695949
## mnth2         -94.736059
## mnth3         138.016527
## mnth4         101.583352
## mnth5         155.246572
## mnth6          52.969314
## mnth7          84.352440
## mnth8          78.392780
## mnth9         187.654502
## mnth10        207.809787
## mnth11          7.652538
## mnth12        -99.071931
## holiday1     -216.286739
## weekday1        .
## weekday2      -58.329490
## weekday3      -63.347543
## weekday4      -50.429412
## weekday5      113.513326
## weekday6      150.114135
## workingday1  -742.053358
## weathersit2   -91.703583
## weathersit3  -223.792809
## temp         4517.048936
## tempSq      -2503.900731
## atemp         161.678691
## atempSq      -815.782907
## hum           745.950510
## humSq       -1137.502892
## windspeed    -399.096915
## windspeedSq -1791.924014
```

```
coef(LassoMod, s = "lambda.1se")
```

```
## 35 x 1 sparse Matrix of class "dgCMatrix"
##                         1
## (Intercept)   497.587204
## (Intercept)     .
```

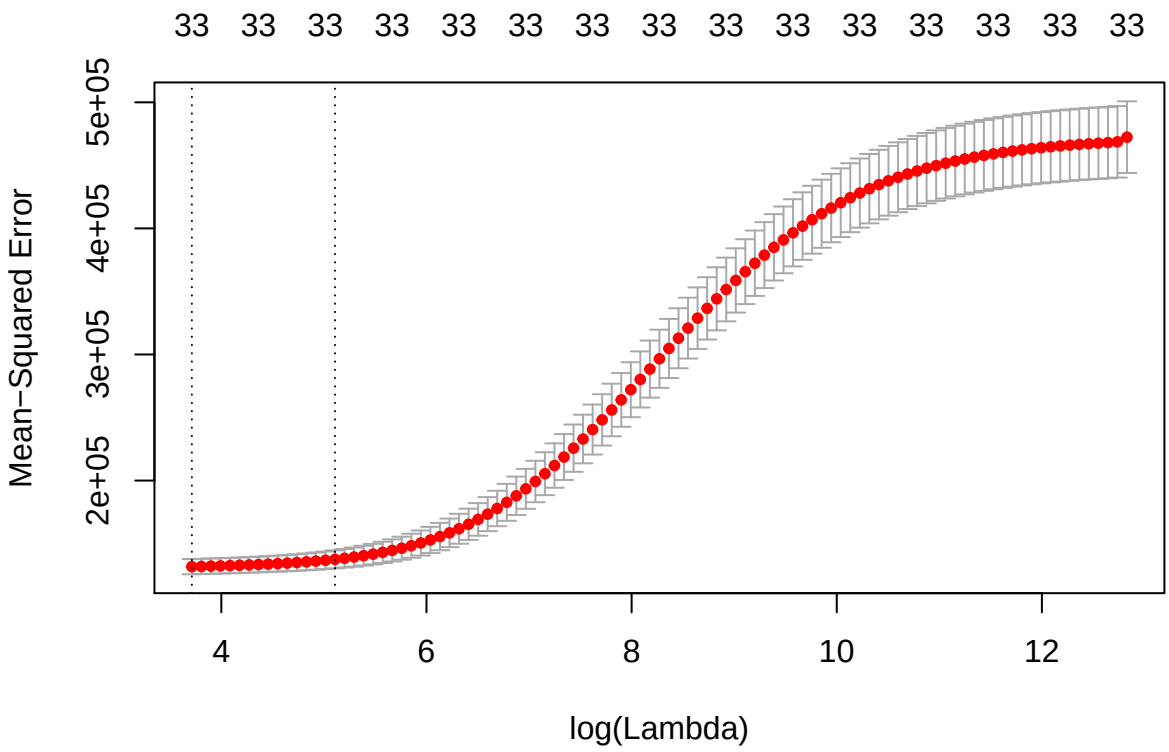
```
## season2       210.474192
## season3         .
## season4         7.009135
## yr1           256.418875
## mnth2        -110.668501
## mnth3          50.100147
## mnth4          30.881539
## mnth5          29.271152
## mnth6           .
## mnth7           .
## mnth8           .
## mnth9         146.414979
## mnth10        179.551072
## mnth11          .
## mnth12        -70.326872
## holiday1     -120.626696
## weekday1        .
## weekday2       -5.237374
## weekday3       -9.527506
## weekday4        .
## weekday5      101.602454
## weekday6      139.716822
## workingday1  -727.753723
## weathersit2   -71.702483
## weathersit3  -213.676670
## temp         1332.182645
## tempSq          .
## atemp         505.982532
## atempSq         .
## hum             .
## humSq        -356.181440
## windspeed    -275.577914
## windspeedSq -1301.599435
```

The difference that appears in these two parameters is that LassoMod with 1se eliminates a lot more x variables where as min does not. #l.

```
r2Lassomin <- LassoMod$glmnet.fit$dev.ratio[which(LassoMod$glmnet.fit$lambda ==
LassoMod$lambda.min)]
r2Lasso1se <- LassoMod$glmnet.fit$dev.ratio[which(LassoMod$glmnet.fit$lambda ==
LassoMod$lambda.1se)]
```

m.

```
RidgeMod <- cv.glmnet(x = xVars, y = yVars , alpha =  0)
plot(RidgeMod)
```



n

```
coef(RidgeMod, s = "lambda.min")
```

```
## 35 x 1 sparse Matrix of class "dgCMatrix"
##                       1
## (Intercept)   538.46722
## (Intercept)     .
## season2       217.89966
## season3        90.72819
## season4        80.07643
## yr1           270.36753
## mnth2         -74.93626
## mnth3         190.45763
## mnth4         182.05297
## mnth5         201.13483
## mnth6          56.53964
## mnth7          11.79217
## mnth8          87.42296
## mnth9         243.49533
## mnth10        265.75957
## mnth11         59.66073
## mnth12        -72.65235
## holiday1      -13.08881
## weekday1     -174.23744
## weekday2     -230.16192
## weekday3     -232.89360
## weekday4     -223.57352
```

```
## weekday5      -57.13574
## weekday6      179.27647
## workingday1  -516.94786
## weathersit2   -98.61518
## weathersit3  -278.24659
## temp          900.16107
## tempSq         44.68471
## atemp         857.72992
## atempSq      -103.73839
## hum          -127.61705
## humSq        -327.20313
## windspeed    -454.27029
## windspeedSq -1335.46001
```

```
coef(RidgeMod, s = "lambda.1se")
```

```
## 35 x 1 sparse Matrix of class "dgCMatrix"
##                       1
## (Intercept)   644.41127
## (Intercept)     .
## season2       176.77014
## season3        56.46997
## season4        67.13920
## yr1           235.63499
## mnth2        -120.94792
## mnth3         125.40987
## mnth4         148.84245
## mnth5         159.69810
## mnth6          30.36425
## mnth7         -22.18367
## mnth8          46.60195
## mnth9         185.51962
## mnth10        195.30477
## mnth11         16.34527
## mnth12       -109.54934
## holiday1       23.51713
## weekday1     -145.59024
## weekday2     -198.06515
## weekday3     -199.99826
## weekday4     -190.03650
## weekday5      -44.63196
## weekday6      219.37619
## workingday1  -442.46550
## weathersit2   -91.72377
## weathersit3  -264.55479
## temp          571.87593
## tempSq        265.59892
## atemp         612.57684
## atempSq       285.27850
## hum          -188.56881
## humSq        -221.47939
## windspeed    -455.20475
## windspeedSq -1129.41387
```

o.

```
r2Ridgemin <- RidgeMod$glmnet.fit$dev.ratio[which(RidgeMod$glmnet.fit$lambda ==
RidgeMod$lambda.min)]
r2Ridge1se <- RidgeMod$glmnet.fit$dev.ratio[which(RidgeMod$glmnet.fit$lambda ==
RidgeMod$lambda.1se)]
```

p.

```
#install.packages("plotmo")
library(plotmo)
```

```
## Loading required package: plotrix
```

```
## Loading required package: TeachingDemos
```

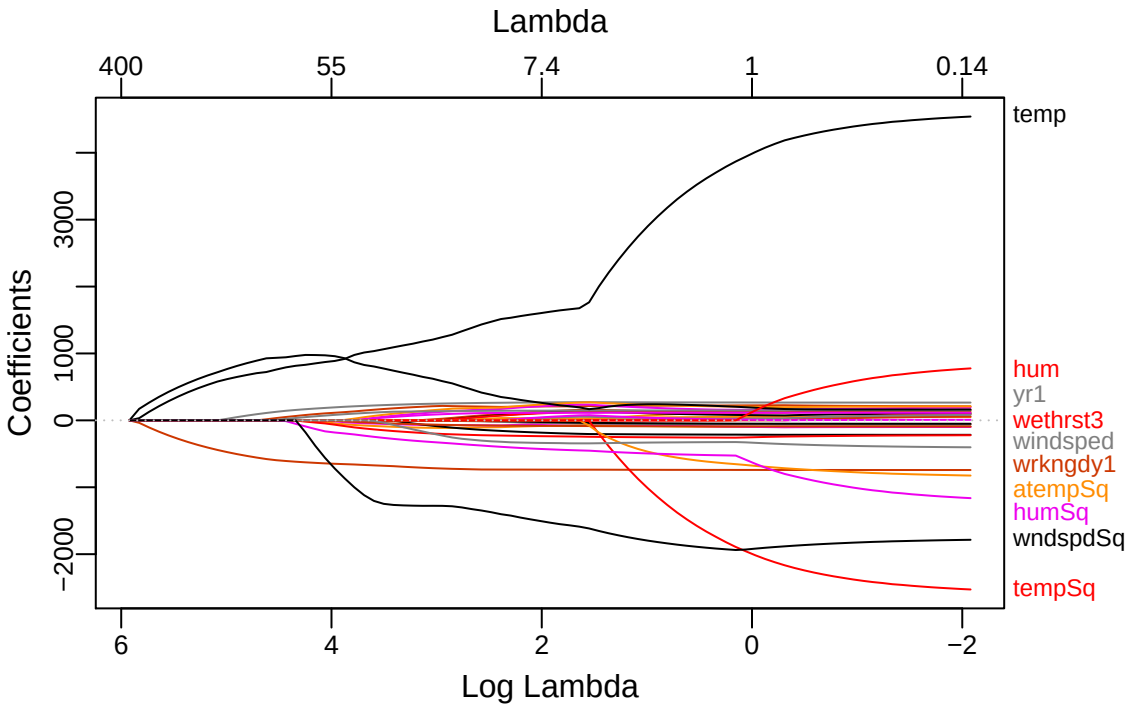
```
LassoMod2 <- glmnet(x = xVars, y = yVars, alpha = 1)
LassoMod2
```

```
##
## Call:  glmnet(x = xVars, y = yVars, alpha = 1)
##
##       Df    %Dev   Lambda
##  [1,]  0 0.00000 373.2000
##  [2,]  3 0.07708 340.0000
##  [3,]  3 0.16530 309.8000
##  [4,]  3 0.23850 282.3000
##  [5,]  3 0.29920 257.2000
##  [6,]  3 0.34970 234.4000
##  [7,]  3 0.39160 213.5000
##  [8,]  3 0.42640 194.6000
##  [9,]  3 0.45520 177.3000
## [10,]  3 0.47920 161.5000
## [11,]  4 0.50640 147.2000
## [12,]  4 0.52990 134.1000
## [13,]  4 0.54950 122.2000
## [14,]  4 0.56570 111.3000
## [15,]  5 0.58180 101.5000
## [16,]  6 0.59620  92.4400
## [17,]  7 0.60930  84.2300
## [18,]  8 0.62210  76.7400
## [19,]  9 0.63510  69.9300
## [20,] 10 0.64700  63.7100
## [21,] 10 0.65700  58.0500
## [22,] 12 0.66630  52.9000
## [23,] 13 0.67470  48.2000
## [24,] 14 0.68250  43.9200
## [25,] 15 0.69120  40.0100
## [26,] 16 0.69840  36.4600
## [27,] 16 0.70430  33.2200
## [28,] 17 0.70940  30.2700
## [29,] 17 0.71410  27.5800
```

```
## [30,] 17 0.71790  25.1300
## [31,] 18 0.72130  22.9000
## [32,] 19 0.72430  20.8600
## [33,] 21 0.72690  19.0100
## [34,] 22 0.72940  17.3200
## [35,] 23 0.73170  15.7800
## [36,] 23 0.73370  14.3800
## [37,] 25 0.73540  13.1000
## [38,] 25 0.73690  11.9400
## [39,] 27 0.73820  10.8800
## [40,] 27 0.73950   9.9120
## [41,] 27 0.74050   9.0310
## [42,] 27 0.74140   8.2290
## [43,] 27 0.74210   7.4980
## [44,] 26 0.74280   6.8320
## [45,] 26 0.74330   6.2250
## [46,] 26 0.74370   5.6720
## [47,] 27 0.74400   5.1680
## [48,] 29 0.74490   4.7090
## [49,] 29 0.74680   4.2910
## [50,] 29 0.74840   3.9090
## [51,] 29 0.74980   3.5620
## [52,] 29 0.75090   3.2460
## [53,] 29 0.75180   2.9570
## [54,] 29 0.75260   2.6950
## [55,] 29 0.75320   2.4550
## [56,] 29 0.75380   2.2370
## [57,] 29 0.75420   2.0380
## [58,] 29 0.75460   1.8570
## [59,] 29 0.75490   1.6920
## [60,] 29 0.75520   1.5420
## [61,] 29 0.75540   1.4050
## [62,] 30 0.75560   1.2800
## [63,] 31 0.75580   1.1660
## [64,] 31 0.75600   1.0630
## [65,] 31 0.75620   0.9684
## [66,] 31 0.75640   0.8824
## [67,] 31 0.75660   0.8040
## [68,] 32 0.75670   0.7326
## [69,] 32 0.75680   0.6675
## [70,] 32 0.75690   0.6082
## [71,] 32 0.75700   0.5542
## [72,] 32 0.75710   0.5049
## [73,] 32 0.75710   0.4601
## [74,] 32 0.75720   0.4192
## [75,] 32 0.75720   0.3820
## [76,] 32 0.75730   0.3480
## [77,] 32 0.75730   0.3171
## [78,] 32 0.75730   0.2889
## [79,] 32 0.75730   0.2633
## [80,] 32 0.75740   0.2399
## [81,] 32 0.75740   0.2186
## [82,] 32 0.75740   0.1992
## [83,] 32 0.75740   0.1815
```

```
## [84,] 32 0.75740   0.1653
## [85,] 32 0.75740   0.1507
## [86,] 32 0.75740   0.1373
## [87,] 32 0.75740   0.1251
```

```
plot_glmnet(LassoMod2)
```



q.

Well simply comparing based on the coefficient estimates it appears that Lasso mod1se eliminates most the coefficients that are deemed to be irrellevent in an estimation where as RidgeMod 1se doesnt eliminate nearly as many.