2550 Problem Set 5

Alyson Singleton

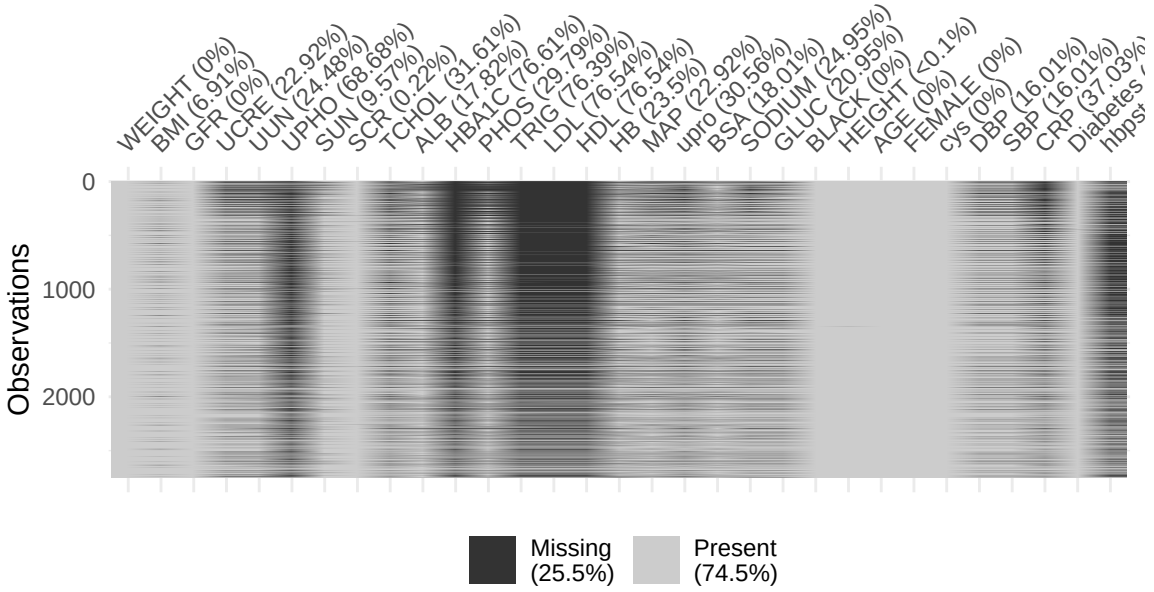
11/22/2019

Introduction

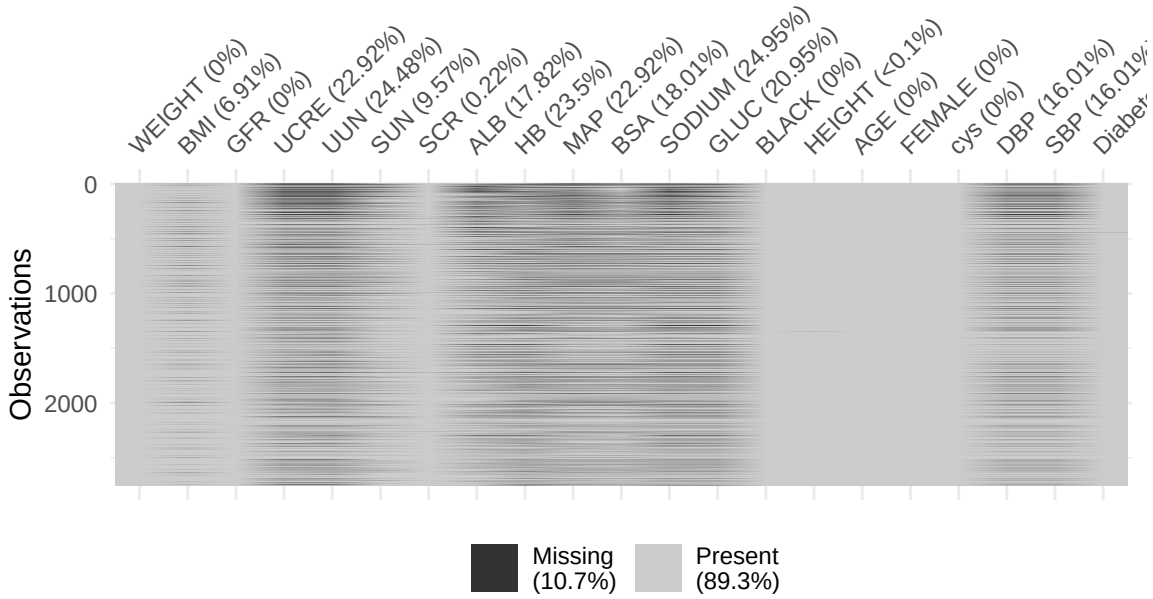
GFR stands for glomerular filtration rate and is considered the best measure of kidney function. We therefore use it as our outcome measure in this analysis to explore possible predictor of kidney disease. Normal kidney function is represented by a GFR of 90 or above [https://www.kidney.org/atoz/content/gfr]. We were directed towards a specfic set of potential predictors in the problem statement and were directed to use the predictors as given. For this reason I did not do any investigation on potential transformation of the predictors or outcome, and also did not consider interaction terms in my analysis. I also did not investigate any potential colinearity as the methods used below to create potential models will attend to this through their variable selection processes. Therefore, the only data exploration that I deemed necessary for this analysis was missing values.

Missing Data

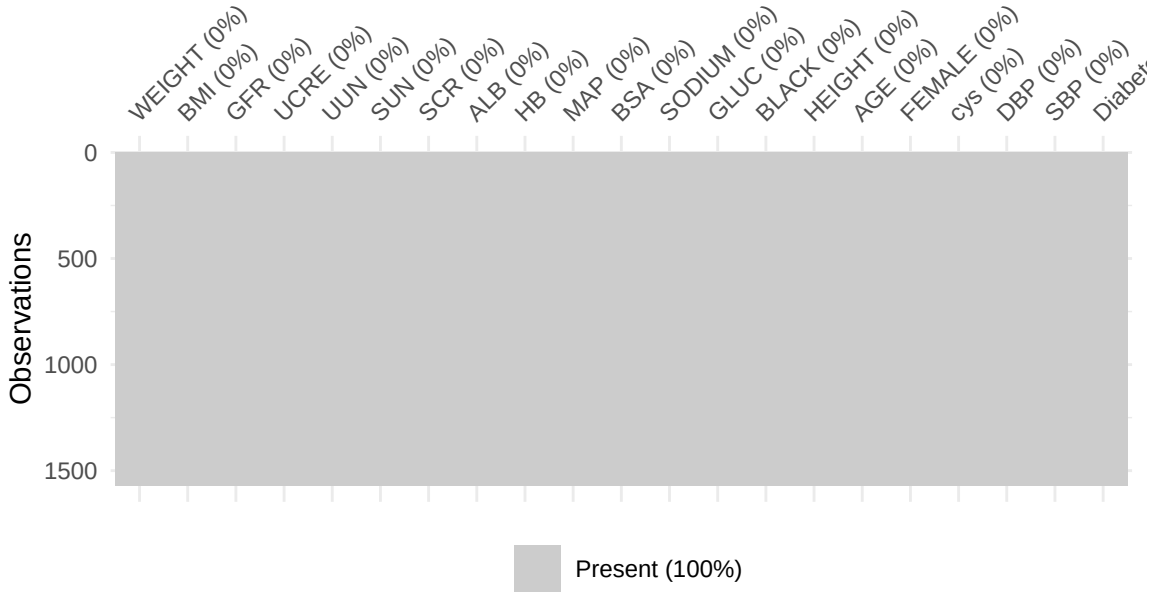
I began by visualizing where the missing data was in the data set.



I decided to remove variables that had more than 30% of their observations missing. This resulted in the removal of HBA1C, TRIG, LDL, HDL, UPHO, hbpstatus, TCHOL, CRP, upro, and PHOS. Without these removals, a reduction down to complete rows would restrict the data set down from 2749 to only 543 rows. See the missingness with these predictors removed below.



Now I restricted the data set to only include observations with completed data across all possible predictors. I recognize that there are other ways to deal with missing data that help you prevent bias and allow you to use the other data included in the row with the missing value. I, however, do not know how to employ these techniques and I opted for explicitly naming their removal and conditioning my results upon this decision. 1568 observations remain for my analysis.



Problem 1

Write a bootstrap algorithm to adjust for optimism on training data in using forward and backward stepwise regression to select a best model for predicting GFR. Carry out the following algorithm.

Part A

Find forward and backward regression to the entire dataset and compute R^2_{train}.

I fit stepwise regression to the data set using both directions in tandem. I used the stepAIC function to do variable selection with the measure of AIC. I stored this as r2.train.

Part B

Bootstrap the data 1000 times and fit forward and backward regression models to each bootstrap, saving $R^2_{boot.train}$.

I wrote a function called "rsq" that fit the stepwise regression model to the bootstrap, again using AIC as our measure, and output the r-squared values associated with the predictions on the training set and those left out in the test set. I ran this on 1000 different boostrapped samples and stored them all in two vectors : train.rsq and test.rsq.

Part C

For each bootstrap sample, calculate $R^2_{boot.test}$ on the part of the original training boot test set not selected into the bootstrap.

Done above in Part B!

Part D

Calculate $R^2_{boot.opt} = R^2_{boot.train} - R^2_{boot.test}$. This is the apparent optimism in the bootstrap sample.

I subtracted the vectors (train.rsq and test.rsq) as indicated and stored them in boot.opt.

Part E

Average $R^2_{boot.opt}$ across the bootstrap samples. This is the average optimism.

I averaged the vector found in Part D and stored it as avg.boot.opt.

Part F

Subtract the average $R^2_{boot.opt}$ from R^2_{train} to get an estimate of R^2_{test}.

I subtracted the average optimism from the R^2_{train} from Part A (r.train) to get an estimate of R^2_{test} (stored as r2.test). All of the R^2 values from Ploblem 1 are included in the summary table below.

Table 1: Summary of R^2 Values in Question 1

	Title	Value
Part A	R^2 Train	0.7171
Part B	Mean R^2 Boot Train	0.7152
Part C	Mean R^2 Boot Test	0.7163
Part E	Mean R^2 Boot Optimism	-0.0011
Part F	R^2 Test	0.7181

Also see below a table of the chosen predictors and their corresponding coefficients.

Table 2: Bootstrap Coefficients

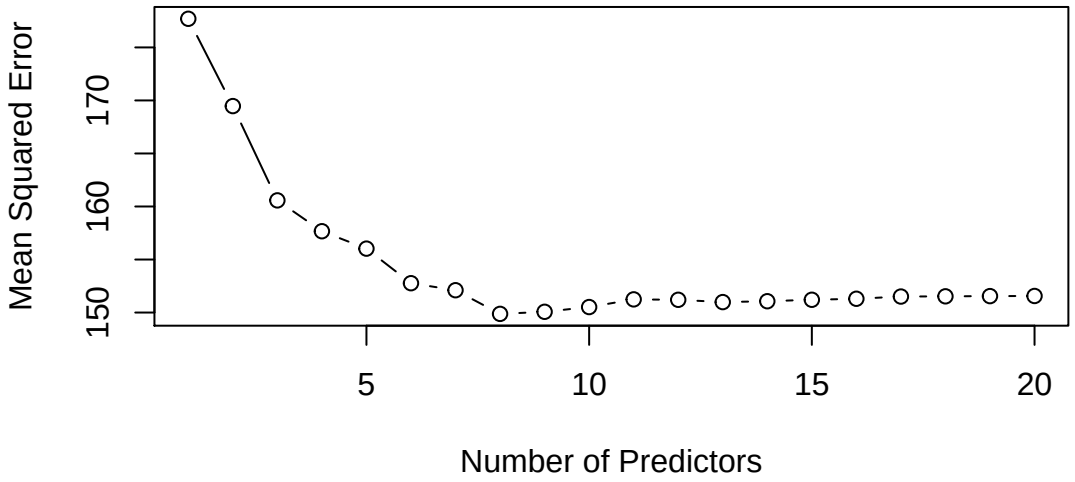
	Coefficient Value
(Intercept)	68.2661165
WEIGHT	-0.0924810
BMI	0.2826924
UCRE	0.0040472
SUN	-0.1717479
SCR	-4.7292218
ALB	3.5739976
BLACK	6.2162064
AGE	-0.1134173
FEMALE	-4.5553533
cys	-13.4258699

Problem 2

Use cross-validation to construct a predictive model with a) stepwise regression; b) ridge regression; c) lasso regression. At the end, refit using the best fitting model of each type on the whole dataset. Compare the estimate of test R2 between the bootstrap approach and the stepwise, ridge and lasso approaches.

Part A - Stepwise Regression

I used k-fold cross-validation to construct a model with stepwise regression. I let $k = 10$ to create 10 total folds to train and test the model on. Below you can see a plot that compares the number of predictors included against the Mean Squared Error (MSE). At first, the MSE decreases as the number of predictors increases. Eventually, however, the MSE begins to increase. It is at this point that we conclude we have optimized the number of predictors to maximize both the predictive ability of our model and its external validity.



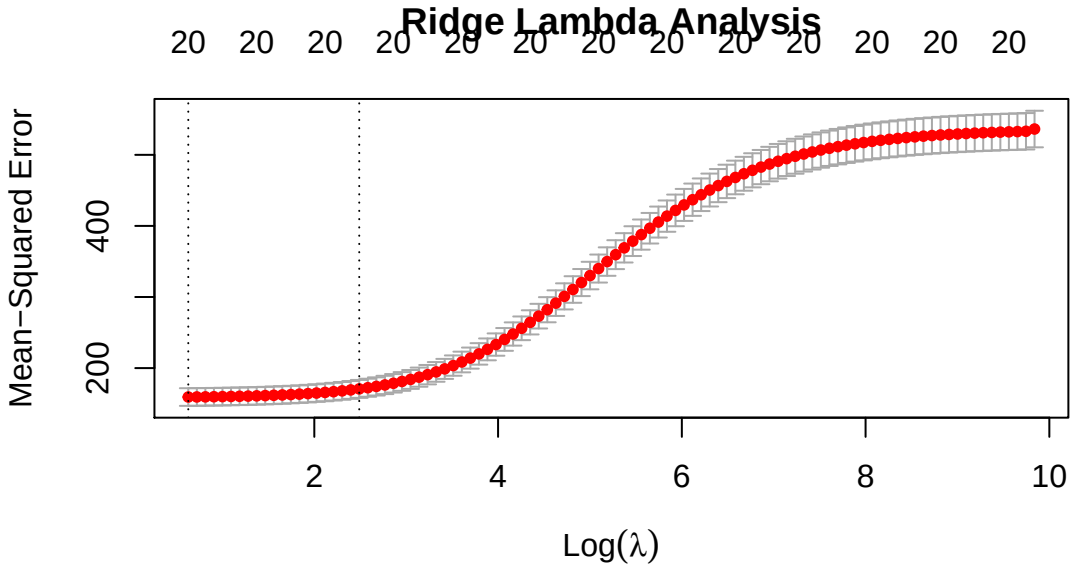
Also see below a table of the chosen predictors and their corresponding coefficients.

Table 3: Stepwise K-Fold Coefficients

	Coefficient Value
(Intercept)	67.9438186
UCRE	0.0037993
SUN	-0.1660853
SCR	-4.7773957
ALB	3.5823222
BLACK	6.3356124
AGE	-0.1037954
FEMALE	-3.4251286
cys	-13.4562585

Part B - Ridge Regression

Next, I used k-fold cross-validation to construct a model with ridge regression. I again let $k = 10$ to create 10 total folds to train and test the model on. Below you can see a plot that compares the log(lambda), a hyperparameter, against the Mean Squared Error (MSE). We choose the lambda that minimizes the mean MSE across the k folds, which ends up being about 1.87.



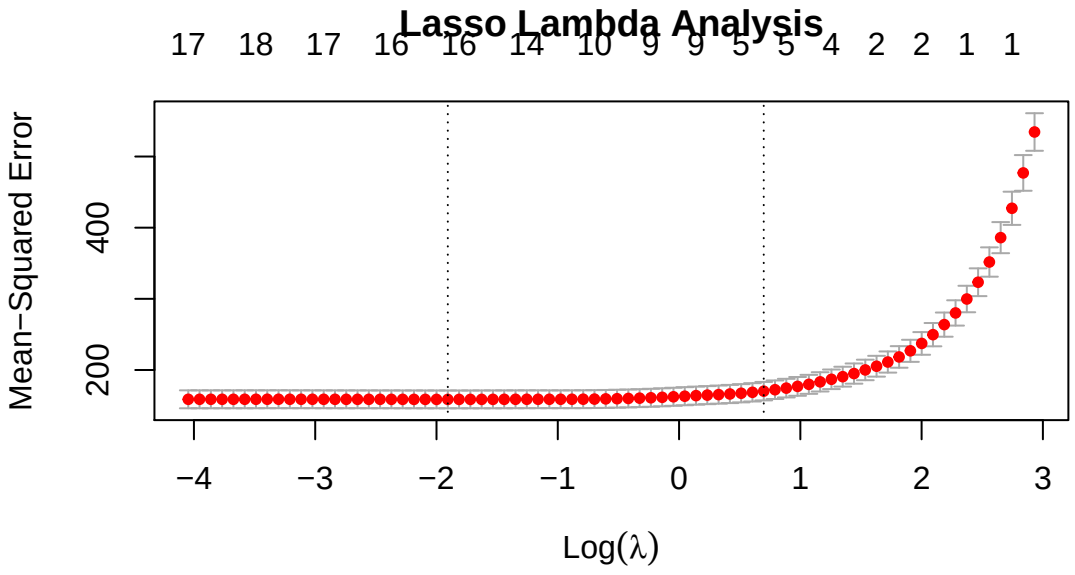
See below a table of the chosen predictors and their corresponding coefficients.

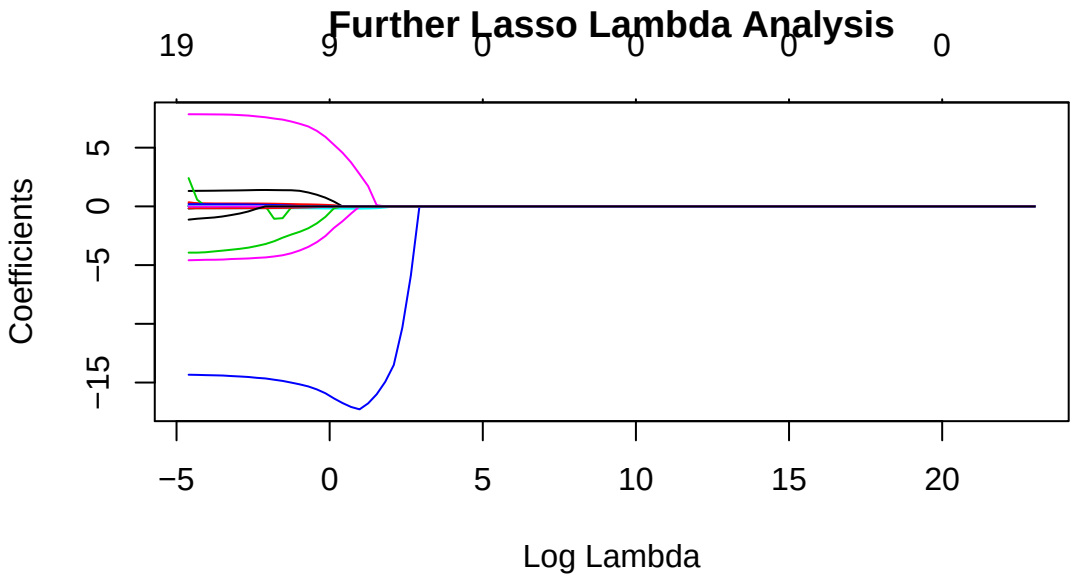
Part C - Lasso Regression

For the final time, I used k-fold cross-validation to construct a model, this time with lasso regression. I again let $k = 10$ to create 10 total folds to train and test the model on. Below you can see another plot that compares the log(lambda), a hyperparameter, against the Mean Squared Error (MSE). We choose the lambda that minimizes the mean MSE across the k folds, which ends up being about 0.14 in this case. You also see a second plot that shows how the lambda value influences how the coefficients of the predictor variables are or are not sent to zero.

Table 4: Ridge K-Fold Coefficients

	Coefficient Value
(Intercept)	50.8544260
WEIGHT	-0.0126305
BMI	0.0585474
UCRE	0.0031402
UUN	0.2120020
SUN	-0.2400534
SCR	-4.8052667
ALB	3.9215734
HB	0.2128133
MAP	0.0030483
BSA	-1.1228321
SODIUM	0.1068303
GLUC	0.0167258
BLACK	6.4152406
HEIGHT	-0.0322306
AGE	-0.1001718
FEMALE	-3.6126765
cys	-11.1165347
DBP	0.0083550
SBP	-0.0053549
Diabetes	0.0068737





Also see below a table of the chosen non-zero predictors and their corresponding coefficients.

Table 5: Lasso K-Fold Coefficients

	Coefficient Value
(Intercept)	62.9153262
UCRE	0.0033529
UUN	0.0961313
SUN	-0.1843719
SCR	-4.1957136
ALB	3.3533793
HB	0.0647795
SODIUM	0.0629703
GLUC	0.0090456
BLACK	6.0768053
HEIGHT	-0.0315432
AGE	-0.0942875
FEMALE	-3.3232803
cys	-13.6499611

Problem 3

Describe the findings of the different models in clearly written text, tables and figures discussing both the differences between the model findings and consistencies. Which factors are predictive? How well do the models predict the outcomes?

See above for explanations and analysis of the figures in this report. Below I will compare the preformance and choices of the models. First, we compare the R^2 values between each of the models as a method to compare their ability to predict GFR as an outcome of their chosen predicts on the entire dataset. All of the models' R^2 values are quite close together! The bootstrap method has the highest R^2, but only over the lasso regression by just over 0.001. After lasso was stepwise and last was ridge. It was interesting to me that, ridge regression, the model with the most variables included by definition, was the least predictive.

Even though it is able to severely minimize the impact of variables it identifies as having less value, I would hypothesize that it does slightly worse when tested on the entire data set because it is slightly overfitting when using all of the variables during training. Ridge also had the highest MSE.

Table 6: Summary of R^2 Values between Questions 1 and 2

	Number of Coefficients	Lambda	R^2 Value	MSE Values
Bootstrap	11	NA	0.71814	NA
Stepwise Regression	9	NA	0.71612	146.7568
Ridge Regression	21	1.87466	0.71564	147.0079
Lasso Regression (non-zero)	14	0.14856	0.71687	146.3707

Table 7: Coefficients Comparison

	Bootstrap	Stepwise	Ridge	Lasso
Intercept	68.2661	67.9438	50.8544	62.9153
WEIGHT	-0.0925	NA	-0.0126	0.0000
BMI	0.2827	NA	0.0585	0.0000
UCRE	0.0040	0.0038	0.0031	0.0034
UUN	NA	NA	0.2120	0.0961
SUN	-0.1717	-0.1661	-0.2401	-0.1844
SCR	-4.7292	-4.7774	-4.8053	-4.1957
ALB	3.5740	3.5823	3.9216	3.3534
HB	NA	NA	0.2128	0.0648
MAP	NA	NA	0.0030	0.0000
BSA	NA	NA	-1.1228	0.0000
SODIUM	NA	NA	0.1068	0.0630
GLUC	NA	NA	0.0167	0.0090
BLACK	6.2162	6.3356	6.4152	6.0768
HEIGHT	NA	NA	-0.0322	-0.0315
AGE	-0.1134	-0.1038	-0.1002	-0.0943
FEMALE	-4.5554	-3.4251	-3.6127	-3.3233
cys	-13.4259	-13.4563	-11.1165	-13.6500
DBP	NA	NA	0.0084	0.0000
SBP	NA	NA	-0.0054	0.0000
Diabetes	NA	NA	0.0069	0.0000

Now let us compare which predictors were selected by which models. See above for all of the predictors selected and their corresponding coefficients. I will only discuss a few here. There were 8 total predictors that were selected by all of the models (and a substantial distance from zero for the ridge regression). These variables included urine creatinine (UCRE), serum urea nitrogen (SUN), serum creatinine (SCR), albumin (ALB), race (BLACK), age (AGE), sex (FEMALE), and serum cystatin (cys). Additionally, the coefficients for all of these predictors had common sign and magnitude across each of the models. This is encouraging to me and comments on the validity of each of the methods. I would conclude that these are likely the predictive factors.

I found it intersting that the bootstrapped model chose to include weight (kg, WEIGHT) and body mass index (BMI), while stepwise and lasso did not. It's coefficients for these variables were much higher than those assigned to it in the ridge regression model as well. These were the only variables that boot included that stepwise did not. We recall that the boot model did have the higher R^2, maybe because of the inclusion of

these two variables. I would possibly include these in the list of predictive factors with further investigation.

Lasso did not include weight or body mass index but did include urine urea nitrogen (UUN), hemoglobin (HB), sodium (SODIUM), glucose (GLUC), and height (cm, HEIGHT). Most of these ended up having coefficients similar to those reported from the ridge regression, the closest between the two being their height coefficients.

Overall, I greatly enjoyed this analysis and the opportunity to compare these methods. In the end, I would choose to use the bootstrapped model. It had the highest R^2 value, and I liked that it represented a middle ground in the number of variables selected as predictors. It interestingly also took me the longest to run!

Code Appendix

```
knitr::opts_chunk$set(echo = F, results='asis', warning=F, message=F, cache=F)
pacman::p_load(visdat)
setwd("~/Desktop/masters/PHP2550/pset5")
library(dplyr)
library(tidyr)
library(reshape2)
library(naniar)
library(grDevices)

#read in data set
ioddata.original <- read.csv("iodatadev.csv")

#separate into training and testing data
#training.wells <- wells[1:2520,]
#testing.wells <- wells[2521:3020,]

#basic outcome investigation
#plot(ioddata$GFR)

# keep only variable names indicated in the problem statement, removal of 26 columns (57 down to 31)
ioddata = subset(ioddata.original, select = c(WEIGHT, BMI, GFR, UCRE, UUN, UPHO, SUN, SCR, TCHOL, ALB, I
#ioddata=na.omit(ioddata)

# missing values investigation
vis_miss(ioddata)
#remove variables w more than 30% missing
ioddata$HBA1C <- NULL
ioddata$TRIG <- NULL
ioddata$LDL <- NULL
ioddata$HDL <- NULL
ioddata$UPHO <- NULL
ioddata$hbpstatus <- NULL
ioddata$TCHOL <- NULL
ioddata$CRP <- NULL
ioddata$upro <- NULL
ioddata$PHOS <- NULL

#ioddata$UCRE <- NULL
#ioddata$UUN <- NULL
```

```r
#ioddata$HB <- NULL
#ioddata$MAP <- NULL
#ioddata$SODIUM <- NULL
#ioddata$GLUC <- NULL
#ioddata$GLUC <- NULL
#ioddata$GLUC <- NULL


vis_miss(ioddata)
# remove all missing vars and their rows--make fully complete data set for simplicity of getting everyth
ioddata=na.omit(ioddata)
vis_miss(ioddata)

## step AIC
library(MASS)
ioddata.lm <- lm(GFR~.,data=ioddata)
ioddata.step <- stepAIC(ioddata.lm, trace = FALSE, direction=c("both"))
#ioddata.step$anova
#ioddata.step$call$formula

ioddata.step.model <- lm(ioddata.step$call$formula, data=ioddata)
s <- summary(ioddata.step.model)

r2.train <- s$r.squared
boot.coefs <- ioddata.step.model$coefficients
library(boot)
set.seed(2)

# build function to make models and grab / output r-squared
rsq = function(data,index) {  # Function to bootstrap model rsq
  ioddata.lm.boot <- lm(GFR~., data=data, subset=index)
  ioddata.step.boot <- stepAIC(ioddata.lm.boot, trace = FALSE, direction=c("both"))

  ioddata.step.model.boot <- lm(ioddata.step.boot$call$formula, data=data, subset=index)
  #print(length(ioddata.step.model$coefficients))
  #print(ioddata.step.model$coefficients)
  summary(ioddata.step.model.boot)$r.squared
}


# initializations
n <- length(ioddata$GFR)
boots <- 1000
train.rsq <- as.vector(rep(0,boots))
test.rsq <- as.vector(rep(0,boots))

for (i in 1:boots) {
  train.i <- sample(n,n,replace=T)
  test.data <- ioddata[-train.i,]
  test.data.i <- dim(test.data)[1]

  train.rsq[i] <- rsq(ioddata,train.i)
  test.rsq[i] <- rsq(test.data,c(1:test.data.i))
```

```
}

avg.boot.train <- mean(train.rsq)
# calculated above
avg.boot.test <- mean(test.rsq)
boot.opt <- train.rsq - test.rsq
avg.boot.opt <- mean(boot.opt)
library(knitr)
library(kableExtra)
library(ggplot2)

r2.test <- r2.train - avg.boot.opt

r2.summary.df <- data.frame(c("R^2 Train", round(r2.train,4)),
                            c("Mean R^2 Boot Train", round(avg.boot.train,4)),
                            c("Mean R^2 Boot Test", round(avg.boot.test,4)),
                            c("Mean R^2 Boot Optimism", round(avg.boot.opt,4)),
                            c("R^2 Test", round(r2.test,4)))
r2.summary.df <- t(r2.summary.df)
colnames(r2.summary.df) <- c("Title","Value")
rownames(r2.summary.df) <- c("Part A",
                             "Part B",
                             "Part C",
                             "Part E",
                             "Part F")

#build table
kable(r2.summary.df, "latex", caption = "Summary of $R^2$ Values in Question 1", booktabs = T) %>%
kable_styling(latex_options = c("striped", "hold_position"))

boot.coefs.df <- data.frame(boot.coefs)

#boot.coefs.df <- t(boot.coefs.df)
colnames(boot.coefs.df) <- c("Coefficient Value")
# rownames(boot.coefs.df)

#build table
kable(boot.coefs.df, "latex", caption = "Bootstrap Coefficients", booktabs = T) %>%
kable_styling(latex_options = c("striped", "hold_position"))
library(boot)
library(leaps)

# take predict.regsubsets function from Variable Selection code for predict function replacement

predict.regsubsets=function(object,newdata,id,...){
  form=as.formula(object$call[[2]])
  mat=model.matrix(form,newdata)
  coefi=coef(object,id=id)
  xvars=names(coefi)
  mat[,xvars]%*%coefi
}

# k-fold cross-validation
```

```r
# initializations

set.seed(1)
k.folds <- 10
variables.n <- dim(ioddata)[2]-1

train <- sample(c(TRUE,FALSE), nrow(ioddata),replace=TRUE)
test <- (!train)

x <- model.matrix(GFR~.,data=ioddata)[,-1]
y <- ioddata$GFR

folds <- sample(1:k.folds,nrow(ioddata),replace=TRUE)
cv.errors <- matrix(NA,k.folds,variables.n, dimnames=list(NULL, paste(1:variables.n)))

# find best fit using all the different fold options
for(j in 1:k.folds){
  best.fit=regsubsets(GFR~.,data=ioddata[folds!=j,],nvmax=variables.n)
  for(i in 1:variables.n){
    pred=predict.regsubsets(best.fit,ioddata[folds==j,],id=i)
    cv.errors[j,i]=mean((ioddata$GFR[folds==j] - pred) ^2)
  }
}

mean.cv.errors=apply(cv.errors,2,mean)
#mean.cv.errors
#par(mfrow=c(1,1))
plot(mean.cv.errors,type='b', xlab= "Number of Predictors", ylab="Mean Squared Error")
hyper.param <- which.min(mean.cv.errors)
#hyper.param
# applies stepwise work to entire data set, rather than to the folds, with the found selection of using

reg.best <- regsubsets(GFR~.,data=ioddata, nvmax=variables.n)
coef.of.best <- coef(reg.best,hyper.param)

#mean.cv.errors[hyper.param]
#cross validation on stepwise regression
#40.91286

# then find rsq
#summary(reg.best)$rsq[hyper.param]

pred.step=predict.regsubsets(reg.best,ioddata,id=hyper.param)

# MSE on entire set
mse.step <- mean((pred.step-y)^2)
#145.7762

# Rsq on entire set
err.step <- pred.step - y
r2.step <- 1 - var(err.step)/var(y)
r2.step <- r2.step[1]
#r2.step
```

```r
step.coefs.df <- data.frame(coef.of.best)

#step.coefs.df <- t(step.coefs.df)
colnames(step.coefs.df) <- c("Coefficient Value")
# rownames(step.coefs.df)

#build table
kable(step.coefs.df, "latex", caption = "Stepwise K-Fold Coefficients", booktabs = T) %>%
kable_styling(latex_options = c("striped", "hold_position"))
library(glmnet)
grid=10^seq(10,-2,length=100)    #Grid of lambda values

#Split into training set and test set to estimate test error
set.seed(1)
train=sample(1:nrow(x), nrow(x)/2)
test=(-train)
y.test=y[test]
ridge.mod=glmnet(x[train,],y[train],alpha=0,lambda=grid)

#Now use CV to find best lambda
set.seed(1)
cv.out=cv.glmnet(x[train,],y[train],alpha=0,nfolds=10)
plot(cv.out, main="Ridge Lambda Analysis")
bestlam=cv.out$lambda.min   #Lambda with minimum MSE
#bestlam
ridge.pred=predict(ridge.mod,s=bestlam,newx=x[test,])

#MSE on test set
#mean((ridge.pred-y.test)^2)
#42.35399

# attempt at ridge regression w our found lambda on the entire data set
out <- glmnet(x,y,alpha=0)
ridge.coefs <- predict(out,type="coefficients",s=bestlam)[0:variables.n+1,]  #ridge coefficients for mo
ridge.pred.end <- predict(out, s=bestlam, newx=x)

# MSE on entire set
mse.ridge <- mean((ridge.pred.end-y)^2)
#145.6533

# Rsq on entire set
err <- ridge.pred.end - y
r2.ridge <- 1-var(err)/var(y)
r2.ridge <- r2.ridge[1]
#r2.ridge


ridge.coefs.df <- data.frame(ridge.coefs)

#step.coefs.df <- t(step.coefs.df)
colnames(ridge.coefs.df) <- c("Coefficient Value")
# rownames(step.coefs.df)
```

```r
#build table
kable(ridge.coefs.df, "latex", caption = "Ridge K-Fold Coefficients", booktabs = T) %>%
kable_styling(latex_options = c("striped", "hold_position"))
lasso.mod=glmnet(x[train,],y[train],alpha=1,lambda=grid)
set.seed(1)
cv.out.lasso=cv.glmnet(x[train,],y[train],alpha=1)
plot(cv.out.lasso, main="Lasso Lambda Analysis")

bestlam.lasso=cv.out.lasso$lambda.min
#bestlam.lasso
lasso.pred=predict(lasso.mod,s=bestlam.lasso,newx=x[test,])

#MSE on test set
#mean((lasso.pred-y.test)^2)


out.lasso=glmnet(x,y,alpha=1,lambda=grid)
lasso.coef=predict(out.lasso,type="coefficients",s=bestlam.lasso)[1:variables.n,]
lasso.coef.non.zero <- lasso.coef[lasso.coef!=0]

# me trying to find the lasso regression w our found lambda on the entire data set
ridge.pred.end.lasso=predict(out.lasso, s=bestlam.lasso, newx=x)

# MSE on entire set
mse.lasso <- mean((ridge.pred.end.lasso-y)^2)
#145.8856

# r2 on entire set
err.lasso <- ridge.pred.end.lasso - y
r2.lasso <- 1-var(err.lasso)/var(y)
r2.lasso <- r2.lasso[1]
#r2.lasso
plot(lasso.mod,xvar="lambda", main="Further Lasso Lambda Analysis")
lasso.coefs.df <- data.frame(lasso.coef.non.zero)

#step.coefs.df <- t(step.coefs.df)
colnames(lasso.coefs.df) <- c("Coefficient Value")
# rownames(step.coefs.df)

#build table
kable(lasso.coefs.df, "latex", caption = "Lasso K-Fold Coefficients", booktabs = T) %>%
kable_styling(latex_options = c("striped", "hold_position"))
r2.comparison.df <- data.frame(c(length(boot.coefs), NA, round(r2.test,5), NA),
                               c(hyper.param+1, NA, round(r2.step,5), round(mse.step,4)),
                               c(length(ridge.coefs), round(bestlam,5), round(r2.ridge,5), round(mse.ri
                               c(length(lasso.coef.non.zero), round(bestlam.lasso,5), round(r2.lasso,5)
r2.comparison.df <- t(r2.comparison.df)
colnames(r2.comparison.df) <- c("Number of Coefficients", "Lambda", "R^2 Value", "MSE Values")
rownames(r2.comparison.df) <- c("Bootstrap",
                                "Stepwise Regression",
                                "Ridge Regression",
                                "Lasso Regression (non-zero)")
```

```r
#build table
kable(r2.comparison.df, "latex", caption = "Summary of $R^2$ Values between Questions 1 and 2", booktabs
  kable_styling(latex_options = c("striped", "hold_position"))
library(kableExtra)
library(knitr)
library(dplyr)
coefficients.comparison.df <- data.frame(
  c(boot.coefs.df[1,1], boot.coefs.df[2,1], boot.coefs.df[3,1], boot.coefs.df[4,1], NA, boot.coefs.df[5
  c(step.coefs.df[1,1], NA, NA, step.coefs.df[2,1], NA, step.coefs.df[3,1], step.coefs.df[4,1], step.co
  c(ridge.coefs.df[1,1], ridge.coefs.df[2,1], ridge.coefs.df[3,1], ridge.coefs.df[4,1],  ridge.coefs.df
  c(lasso.coefs.df[1,1], 0, 0, lasso.coefs.df[2,1], lasso.coefs.df[3,1], lasso.coefs.df[4,1], lasso.coe

coefficients.comparison.df <- round(coefficients.comparison.df, 4)

colnames(coefficients.comparison.df) <- c("Bootstrap",
                                          "Stepwise",
                                          "Ridge",
                                          "Lasso")
rownames(coefficients.comparison.df) <- c("Intercept",
                                          "WEIGHT",
                                          "BMI",
                                          "UCRE",
                                          "UUN",
                                          "SUN",
                                          "SCR",
                                          "ALB",

                                          "HB",
                                          "MAP",
                                          "BSA",
                                          "SODIUM",
                                          "GLUC",
                                          "BLACK",
                                          "HEIGHT",
                                          "AGE",
                                          "FEMALE",
                                          "cys",
                                          "DBP",
                                          "SBP",
                                          "Diabetes")
kable(coefficients.comparison.df, "latex", caption = "Coefficients Comparison", booktabs = T) %>%
kable_styling(latex_options = c("striped", "hold_position"))
```