Mini Project 4: Report

Raheel Ahmed

Rs170130

Section 1

Question 1

1. **Given that we are interested in comparing means, perform an exploratory analysis of the data. Which graphic is more appropriate for the task at hand: side-by-side box plots of observations from the two methods or a box plot of difference in observations from the two methods?**  
   There are 72 patients in the oxygen saturation data. Each column corresponds to one of the two different methods for recording oxygen saturation data. Each row provides oxygen saturation data using the two methods for ONE individual patient. Additionally, a boxplot of difference in observations from the two methods (different columns for the same set of rows) will be a more useful metric of evaluation in examining the difference in population means for the two methods.
2. **Provide a point estimate ˆθ of θ, appropriate estimates of bias and standard error of the estimate, and a 95% confidence interval for θ. Interpret the results. (Before doing this exercise, you may want to refresh your memory about, e.g., the difference between a two-sample t-test and a paired t-test.)**
   1. Point estimate of ˆθ: -0.4125
   2. Bias of estimate: 9.65894e-15 (Essentially 0)
   3. Standard error of the estimate: 0.1427011
   4. 95% confidence interval for θ: (-0.5552011, -0.2697989)

It appears that, in general, the oxygen saturation measured by method 1 (pulse oximetry) falls 0.4125 behind method 2 (osm). The estimates for an observation neither tend to overshoot nor tend to undershoot in terms of the difference in values between the two methods. The standard error allows us to confirm with 95% confidence that the difference in value between the two methods lies somewhere between -0.5552011 and -0.2697989.

1. **Write your own code to compute (nonparametric) bootstrap estimates of bias, standard error of ˆθ, and a 95% confidence interval for θ computed using the percentile method. How do these results compare with those in (b)?**
   1. Mean estimate of ˆθ: -0.415675
   2. Bias of estimate: -0.003175
   3. Standard error of the estimate: 0.0157955
   4. 95% confidence interval for θ: (-0.6750347, -0.1513542)

Compared to b, we can see that the mean estimate is very similar. The bias found via bootstrapping does indicate that estimates tend to undershoot the difference in values found between the two methods but barely; it appears very close to 0 just like the value found for b. Similarly, the standard error is very close to what it was in part b. The confidence interval covers the same interval found in b but has a wider range of values that are required for 95% confidence.

1. **Repeat the computation in (c) using boot package and make sure the results match.**
   1. Mean estimate of ˆθ: -0.4125
   2. Bias of estimate: 0.002316667
   3. Standard error of the estimate: 0.01684293
   4. 95% confidence interval for θ: (-0.6944, -0.1278)

The computation using the boot package is extremely similar to our own bootstrap version. The mean estimate is the same as in part b. The bias for b, c, and d are all close to 0. The standard error is similar in value ~ 0.015. The 95% confidence interval for θ covers a larger interval like in part c but is still very close to the original confidence interval for the difference in values for the two methods.

1. **What would you conclude about the population means of the two methods?**I would conclude that they inform us that the pulse oximeter reads tend to report lower oxygen saturation than oxygen saturation monitors. Additionally, I would report that the difference in the values of the methods estimations do not usually overshoot or undershoot in value. I feel like, in comparison to the large range (70-100) that oxygen saturation values can be reported within, that the standard error is minimal and that there is no very significant difference in the measurements provided by either method.

Question 2

1. **Fit a linear regression model using all predictors and compute its test MSE.**  
   Shown in code, with test MSE = 0.8189307 without bias adjustment and 0.8167796 with bias adjustment.
2. **Use best-subset selection based on adjusted R2 to find the best linear regression model. Compute the test MSE of the best model.**  
   The best adjusted R2 value was found for the best model with 4 predictors. The test MSE of the best model was: 0.6145249 without bias adjustment and 0.6141675 with bias adjustment.
3. **Use forward stepwise selection based on adjusted R2 to find the best linear regression model. Compute the test MSE of the best model.**  
   The best adjusted R2 value was found for the best model with 4 predictors. The test MSE of the best model was: 0.6145249 without bias adjustment and 0.6141675 with bias adjustment.
4. **Use backward stepwise selection based on adjusted R2 to find the best linear regression model. Compute the test MSE of the best model.**The best adjusted R2 value was found for the best model with 4 predictors. The test MSE of the best model was: 0.6145249 without bias adjustment and 0.6141675 with bias adjustment.
5. **Use ridge regression with penalty parameter chosen optimally via LOOCV to fit a linear regression model. Compute the test MSE of the model.**  
   Shown in code, with estimated test MSE = 0.5996864
6. **Use lasso with penalty parameter chosen optimally via LOOCV to fit a linear regression model. Compute the test MSE of the model.**  
   Shown in code, with estimated test MSE = 0.5522568
7. **Make a tabular summary of the parameter estimates and test MSEs from (a) - (f). Compare the results. Which model(s) would you recommend?**

|  |  |  |
| --- | --- | --- |
| Models | Parameter estimates | Test MSEs |
| Full |  | 0.8189307 |
| Best SS |  | 0.6145249 |
| Forward SS |  | 0.6145249 |
| Backward SS |  | 0.6145249 |
| Ridge Reg | (Intercept) -0.029640992  Cancervol 0.032075764  weight 0.001150147  age 0.004362445  benpros 0.035502480  vesinv1 0.442132423  capspen 0.029310835  gleason 0.247425068 | 0.5996864 |
| Lasso | (Intercept) -1.899770e-01  cancervol 5.871548e-02  weight 7.861665e-05  age .  benpros 6.037468e-02  vesinv1 5.641867e-01  capspen .  gleason 2.877950e-01 | 0.5522568 |

I would recommend using a model based off of ridge regression or Lasso, although it is good to try all the methods out, in this case, we can see that stepwise selection and best subset selection come up with the same best model with the best adjusted R^2 value and end up having the same LOOCV due to formulating the same ideal models. In this case, it appears that lasso and ridge regression both shrink the unnecessary features towards 0. The continuous nature of this shrinking allows for both optimization and for development of reasonable models with estimated test MSEs that were noticeably lower as well.

Question 3

1. **Fit a logistic regression model using all predictors and compute its test error rate.**  
   Shown in code, with an estimated test error rate of 0.249000 without bias adjustment and 0.248446 with bias adjustment.
2. **Use best-subset selection based on AIC to find the best logistic regression model. Compute the test error rate of the best model.**  
   Note: I used fewer than all predictors for my best-subset selection. Bestglm requires 15 or fewer predictors and 21 predictors took a really long time to even attempt to run, so I selected commonly useful features from my other stepwise selection models. The test error rate I found was 0.240000 without bias adjustment and 0.238906 with bias adjustment.
3. **Use forward stepwise selection based on AIC to find the best logistic regression model. Compute the test error rate of the best model.**  
   Using forward stepwise selection based on AIC, I found that the test error rate of the logistic regression model was 0.240000 without bias adjustment and 0.238906 with bias adjustment.
4. **Use backward stepwise selection based on AIC to find the best logistic regression model. Compute the test error rate of the best model.**Using backward stepwise selection based on AIC, I found that the test error rate of the logistic regression model was 0.240000 without bias adjustment and 0.238906 with bias adjustment.
5. **Use ridge regression with penalty parameter chosen optimally via LOOCV to fit a logistic regression model. Compute the test error rate of the model.**Test error rate was 0.1532243.
6. **Use lasso with penalty parameter chosen optimally via LOOCV to fit a logistic regression model. Compute the test error rate of the model.**  
   Test error rate was 0.1521238.
7. **Make a tabular summary of the parameter estimates and test error rates from (a) - (f). Compare the results. Which model(s) would you recommend? How does this recommendation compare with what you recommended in Mini Project 3?**

|  |  |  |
| --- | --- | --- |
| Models | Parameter estimates | Test MSEs |
| Full |  | 0.249000 |
| Best SS |  | 0.240000 |
| Forward SS |  | 0.240000 |
| Backward SS |  | 0.240000 |
| Ridge Reg |  | Using MSE: 0.1532243 |
| Lasso |  | Using MSE: 0.1521238 |

Similarly to our last attempt at feature subset selection, we notice that stepwise models often yield the same model when using the same metric of evaluation, even exhaustively. I would again recommend that a person try all of these approaches, but in this case, it appears that ridge regression and lasso are ideal for model selection. They both have low MSE and the continuous nature of their shrinking help optimize better than the step-wise nature of the other approaches in this case.  
  
This recommendation is different from what I formulated in Mini Project 3 in terms of the number of features. Both the ridge regression and lasso models had many more features used compared to what I selected in the other project. However, both of these models and my custom model did share many features that were significant, such as checkingstatus1, history, duration, savings, installment, and otherplans.

Code Portion

Question 1

library**(**boot**)**

# Obtain oxygen monitoring data from the oxygen saturation txt file

o2\_data **<-** read.table**(**"oxygen\_saturation.txt", header **=** T, sep**=**"\t"**)**

# Obtain data from pulse oximeter and oxygen saturation monitor separately

pos\_data **<-** o2\_data**[**, 1**]**

osm\_data **<-** o2\_data**[**, 2**]**

# Find the vector of differences between the two methods per individual patient

diff\_data **<-** pos\_data **-** osm\_data

# See the number of patients and the

# distribution of values found using the two methods

nrow**(**o2\_data**)**

hist**(**pos\_data**)**

hist**(**osm\_data**)**

# Print out the mean values for pulse oximeter readings

# and oxygen saturation monitors separately

mean**(**pos\_data**)**

mean**(**osm\_data**)**

# Point estimate of the difference in means between the two methods

pt\_estim\_mean **<-** mean**(**pos\_data**)** **-** mean**(**osm\_data**)**

# Bias of the vector produced from looking at the difference in methods per patient (diff\_data)

mean**(**diff\_data**)** **-** pt\_estim\_mean

# Standard error of diff\_data

st\_err **<-** sd**(**diff\_data**)** **/** sqrt**(**72**)**

# Confidence interval via percentile approach: quantile(diff\_data, c(0.025, 0.975))

# 95% confidence interval of diff\_data

c**(**mean**(**diff\_data**)** **-** st\_err, mean**(**diff\_data**)** **+** st\_err**)**

# Number of entries

n **<-** nrow**(**o2\_data**)**

# number of resamples

b **<-** 1000

# Track estimates of difference of means

estimates **<-** c**()**

**for(**i **in** 1**:**b**){**

# For each resample, sample from 1 to n with replacement

indices\_sampled **<-** sample**(**1**:**n, n, replace **=** T**)**

# Use these indices to select rows to calculate the difference of means from

estimates **<-** c**(**estimates,

mean**(**pos\_data**[**indices\_sampled**])** **-** mean**(**osm\_data**[**indices\_sampled**])**

**)**

**}**

# Metrics you can evaluate

# Mean estimate of the difference of values between the two methods

mean**(**estimates**)**

# Variance found via bootstrapping

var**(**estimates**)**

# Bias found via bootstrapping

mean**(**estimates**)** **-** pt\_estim\_mean

# Standard Error found via bootstrapping

sd**(**estimates**)** **/** sqrt**(**72**)**

# Confidence Interval (via percentile approach)

quantile**(**estimates, c**(**.025, .975**))**

# Establish difference of means function

mean.fn **<-** **function(**x, indices**)** **{**

# Return difference of means over two methods for certain indices

result **<-** mean**(**x**[**indices,1**])** **-** mean**(**x**[**indices,2**])**

return**(**result**)**

**}**

# Test the function for the original data

mean.fn**(**o2\_data, 1**:**nrow**(**o2\_data**))**

# Conduct a boostrapping using the boot package.

mean.boot **<-** boot**(**o2\_data, mean.fn, R **=** 1000**)**

mean.boot

# Estimate for difference of means

mean.boot**$**t0

# Bias for estimate of difference of means

mean**(**mean.boot**$**t**)** **-** mean.boot**$**t0

# Standard errorof the estimate of difference of means

sd**(**mean.boot**$**t**)** **/** sqrt**(**72**)**

# Confidence intervals produced with 95% levels

boot.ci**(**mean.boot, type **=** "perc"**)**

Question 2

library**(**boot**)**

library**(**leaps**)**

library**(**glmnet**)**

# Read in prostate cancer data

pc\_data **<-** read.csv**(**"prostate\_cancer.csv"**)**

# Eliminate subject number feature

pc\_data **<-** pc\_data**[**,**-**1**]**

# Treat vesinv as a qualitative variable

pc\_data**$**vesinv **<-** factor**(**pc\_data**$**vesinv, order**=**F, levels **=** c**(**0, 1**))**

# Conduct a natural log transformation on the response

# to adjust it's distribution to something more appropriate.

pc\_data**[**, 1**]** **<-** log**(**pc\_data**[**, 1**])**

hist**(**pc\_data**[**, 1**])**

# Make a full model of a linear regression with psa as response and all features as predictors

# Calculate test MSE via LOOCV

full\_model **<-** glm**(**psa **~** ., data **=** pc\_data**)**

cv.err **<-** cv.glm**(**pc\_data, full\_model**)**

cv.err**$**delta

# Find a reasonable subset of features to implement a linear regression model with

# via the best-subset selection accounting for the best adjusted R^2.

regfit\_full **=** regsubsets**(**psa **~** ., data **=** pc\_data, nvmax **=** 7**)**

regfit\_summ **<-** summary**(**regfit\_full**)**

which.max**(**regfit\_summ**$**adjr2**)**

coef**(**regfit\_full, 4**)**

# Find the test MSE via LOOCV

cv.glm**(**pc\_data, glm**(**psa **~** cancervol **+** benpros **+** vesinv **+** gleason, data**=**pc\_data**))$**delta

# Find a reasonable subset of features to implement a model with using forward

# subset selection with best adjusted R^2 value.

fit.fwd **=** regsubsets**(**psa **~** ., data **=** pc\_data, nvmax **=** 7, method **=** "forward"**)**

summary**(**fit.fwd**)**

which.max**(**summary**(**fit.fwd**)$**adjr2**)**

coef**(**fit.fwd, 4**)**

cv.glm**(**pc\_data, glm**(**psa **~** cancervol **+** benpros **+** vesinv **+** gleason, data**=**pc\_data**))$**delta

# Find a reasonable subset of features to implement a model with using backward

# subset selection with best adjusted R^2 value.

fit.bwd **=** regsubsets**(**psa **~** ., data **=** pc\_data, nvmax **=** 7, method **=** "backward"**)**

summary**(**fit.bwd**)**

which.max**(**summary**(**fit.bwd**)$**adjr2**)**

coef**(**fit.bwd, 4**)**

cv.glm**(**pc\_data, glm**(**psa **~** cancervol **+** benpros **+** vesinv **+** gleason, data**=**pc\_data**))$**delta

# Select response and feature data as y and X respectively

y **<-** pc\_data**$**psa

X **<-** model.matrix**(**psa **~** ., pc\_data**)[**, **-**1**]**

# Set up a grid of potential lambda values

grid **<-** 10**^**seq**(**10, **-**2, length **=** 100**)**

# Using alpha = 0, conduct a ridge regression

ridge.mod **<-** glmnet**(**X, y, alpha **=** 0**)**

# Use LOOCV to determine the best penalty parameter

cv\_ridge **<-** cv.glmnet**(**X, y, alpha **=** 0**)**

best\_ridge\_lambda **<-** cv\_ridge**$**lambda.min

best\_ridge\_lambda

# Use the best lambda to find the best ridge regression model

best\_ridgemod **<-** glmnet**(**X, y, alpha **=** 0, lambda **=** best\_ridge\_lambda**)**

coef**(**best\_ridgemod**)**

# Predict the values using the best ridge regression model and find the MSE

y\_pred **<-** predict**(**ridge.mod, s **=** best\_ridge\_lambda, newx **=** X**)**

mean**((**y\_pred **-** y**)^**2**)**

# Using alpha=1, conduct lasso

lasso.mod **<-** glmnet**(**X, y, alpha**=**1**)**

# Use LOOCV to find the best penalty parameter

cv\_lasso **<-** cv.glmnet**(**X, y, alpha**=**1**)**

best\_lasso\_lambda **<-** cv\_lasso**$**lambda.min

best\_lasso\_lambda

# Make the best lasso model with the best lambda value

best\_lassomod **<-** glmnet**(**X, y, alpha**=**1, lambda**=**best\_lasso\_lambda**)**

coef**(**best\_lassomod**)**

# Predict the appropriate values using the best lasso model and calculate MSE

y\_pred **<-** predict**(**lasso.mod, s **=** best\_lasso\_lambda, newx **=** X**)**

mean**((**y\_pred**-**y**)^**2**)**

Question 3

library**(**boot**)**

library**(**MASS**)**

library**(**glmulti**)**

library**(**glmnet**)**

# Read in german credit dataset and store as a dataframe

german\_data **<-** read.csv**(**"germancredit.csv", stringsAsFactors **=** T**)**

# Produce a logistic regression model using all the predictors

all\_lr **<-** glm**(**Default **~** ., data **=** german\_data, family **=** "binomial"**)**

summary**(**all\_lr**)**

coef**(**all\_lr**)**

# Produce a logistic regression model using no predictors (yields intercept only model)

empty\_lr **<-** glm**(**Default **~** 1, data**=**german\_data, family **=** "binomial"**)**

summary**(**empty\_lr**)**

# Use boot package to estimate LOOCV for log-reg models

# Make cost function

cost **<-** **function(**r, pi **=** 0**){**mean**(**abs**(**r **-** pi**)** **>** 0.5**)}**

# Calculate LOOCV for both the full model and the null model

all\_lr.err **<-** cv.glm**(**german\_data, all\_lr, cost, K**=**nrow**(**german\_data**))**

all\_lr.err**$**delta

empty\_lr.err **<-** cv.glm**(**german\_data, empty\_lr, cost, K**=**nrow**(**german\_data**))**

empty\_lr.err**$**delta

# Find best subset selection model with fewer predictors to improve run time.

# AIC is our method of evaluation, run exhaustive checks for the best model.

best\_sub **<-** glmulti**(**Default **~** checkingstatus1 **+** duration **+** history **+** purpose **+** savings **+**

others **+** installment **+** status **+** amount **+** otherplans **+** foreign **+**

age **+** housing **+** tele, data**=**german\_data, level**=**1,

method**=**"h", crit**=**"aic", confsetsize **=** 2,

plotty**=**F, report**=**F,

fitfunction **=** "glm", family**=**binomial**)**

# Plug in the formula to the glm function and run a logistic regression

best\_lr **<-** glm**(**best\_sub@formulas**[[**1**]]**, data**=**german\_data, family**=**"binomial"**)**

summary**(**best\_lr**)**

coef**(**best\_lr**)**

# Print LOOCV estimate of test error.

best\_lr.err **<-** cv.glm**(**german\_data, best\_lr, cost, K**=**nrow**(**german\_data**))**

best\_lr.err**$**delta

# Find forward stepwise selection model with AIC as our method of evaluation

# I started with the null model and progressed forward with a specified scope

forward\_step **<-** stepAIC**(**empty\_lr,

scope **=** list**(**lower**=**empty\_lr, upper**=**all\_lr**)**,

direction **=** "forward"**)**

# Feed the formula found into the glm function and conduct a logistic regression

forward\_lr **<-** glm**(**Default **~** checkingstatus1 **+** duration **+** history **+** purpose **+** savings **+**

others **+** installment **+** status **+** amount **+** otherplans **+** foreign **+**

age **+** housing **+** tele, data **=** german\_data, family**=**"binomial"**)**

summary**(**forward\_lr**)**

coef**(**forward\_lr**)**

# Calculate LOOCV estimate of test error

forward\_lr.err **<-** cv.glm**(**german\_data, forward\_lr, cost, K**=**nrow**(**german\_data**))**

forward\_lr.err**$**delta

# Find backward stepwise selection model with AIC as our method of evaluation

# I started with the full model and moved backward with the specified scope

backward\_step **<-** stepAIC**(**all\_lr,

scope **=** list**(**lower**=**empty\_lr, upper**=**all\_lr**)**,

direction **=** "backward"**)**

# Feed the formula found into the glm function and conduct a logistic regression

backward\_lr **<-** glm**(**Default **~** checkingstatus1 **+** duration **+** history **+** purpose **+** amount **+**

savings **+** installment **+** status **+** others **+** age **+** otherplans **+**

housing **+** tele **+** foreign, data **=** german\_data, family **=** "binomial"**)**

summary**(**backward\_lr**)**

coef**(**backward\_lr**)**

# Calculate LOOCV estimate of test error

backward\_lr.err **<-** cv.glm**(**german\_data, backward\_lr, cost, K**=**nrow**(**german\_data**))**

backward\_lr.err**$**delta

# Assign response and feature matrix to y and X respectively

y **<-** german\_data**$**Default

X **<-** model.matrix**(**Default **~** ., german\_data**)[**, **-**1**]**

# Set up a grid of potential lambda values

grid **<-** 10**^**seq**(**10, **-**2, length **=** 100**)**

# Conduct a ridge regression where alpha=0

ridge.mod **<-** glmnet**(**X, y, alpha **=** 0**)**

# Use LOOCV to determine the best penalty parameter

cv\_ridge **<-** cv.glmnet**(**X, y, alpha **=** 0**)**

best\_ridge\_lambda **<-** cv\_ridge**$**lambda.min

best\_ridge\_lambda

# Make the best ridge regression model using the best lambda

best\_ridgemod **<-** glmnet**(**X, y, alpha **=** 0, lambda **=** best\_ridge\_lambda**)**

coef**(**best\_ridgemod**)**

# Predict the values of the training data using the best ridge regression model

y\_pred **<-** predict**(**ridge.mod, s **=** best\_ridge\_lambda, newx **=** X**)**

# Compare the values to the true values to find test error

mean**((**y\_pred **-** y**)^**2**)**

# Conduct a lasso model with alpha=1

lasso.mod **<-** glmnet**(**X, y, alpha**=**1**)**

# Use LOOCV to find the best lambda

cv\_lasso **<-** cv.glmnet**(**X, y, alpha**=**1**)**

best\_lasso\_lambda **<-** cv\_lasso**$**lambda.min

best\_lasso\_lambda

# Formulate the best lasso model with the best lasso lambda

best\_lassomod **<-** glmnet**(**X, y, alpha**=**1, lambda**=**best\_lasso\_lambda**)**

coef**(**best\_lassomod**)**

# Predict the values of our training data and compare to the true values to find the test error

y\_pred **<-** predict**(**lasso.mod, s **=** best\_lasso\_lambda, newx **=** X**)**

mean**((**y\_pred**-**y**)^**2**)**