# BIOST 537 Problem Set3

Ivy Zhang

2/9/2022

## Problem 1

### 1(a)

This data contains 30 predictors with a sample size of 569. We have 357 observations in Benign class and 212 observations in Malignant class.

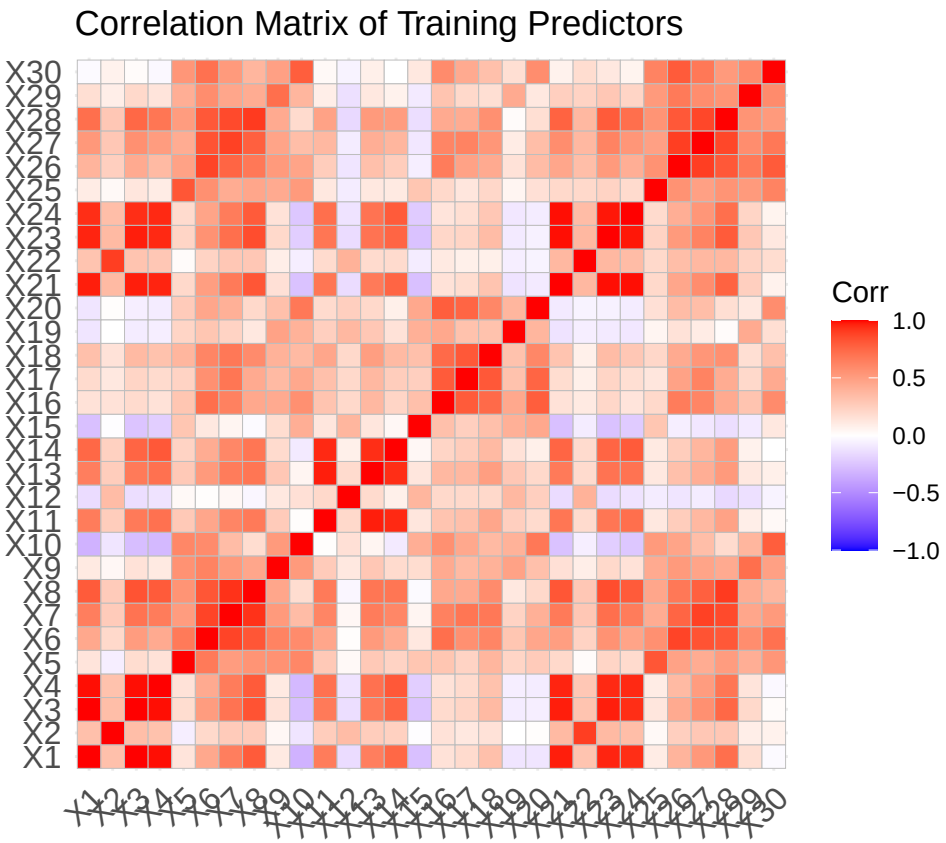
### 1(b)

Process will be shown in the appendix code section.

### 1(c)

Normalize process will be shown in the appendix code section. The reason why we want to perform this step seprately in the training set and test set is because we want these two sets should be normalized independently from each other. We don't want to make the test set normalized affected by the training set, therefore we can have test performance to be more accurate since it should not be biased by the training set.

## 1(d)




From the previous plot, we can clearly see, X1, X3, X4, X20, X23 and X24 are highly correlated with almost correlation equals to 1. Besides these variables, there are some variables are relatively high correlated (e.g: X8 vs X28, X7 as X8). It means we may have collinearity of predictors problem in our data. Our model using all variables may multiple least squares solutions and will also affect other predictors estimation.

## 1(e)

Table 1: Estimated Coefficients of Simple Logistic Regression Using All Predictors

| | Estimate | Std. Error | z value | Pr(>\|z\|) |
|---|---|---|---|---|
| (Intercept) | 92.461 | 28281.573 | 0.003 | 0.997 |
| X1 | -862.997 | 462270.497 | -0.002 | 0.999 |
| X2 | -3.798 | 13416.446 | 0.000 | 1.000 |
| X3 | 989.130 | 544553.105 | 0.002 | 0.999 |
| X4 | -189.766 | 100538.360 | -0.002 | 0.998 |
| X5 | 117.778 | 15574.736 | 0.008 | 0.994 |
| X6 | -302.085 | 41791.183 | -0.007 | 0.994 |
| X7 | -8.086 | 39076.942 | 0.000 | 1.000 |
| X8 | 265.443 | 49259.173 | 0.005 | 0.996 |
| X9 | -122.564 | 22339.352 | -0.005 | 0.996 |
| X10 | 96.252 | 28237.700 | 0.003 | 0.997 |
| X11 | 559.966 | 61288.543 | 0.009 | 0.993 |

| | Estimate | Std. Error | z value | Pr(>\|z\|) |
|---|---|---|---|---|
| X12 | 3.348 | 9628.505 | 0.000 | 1.000 |
| X13 | -110.485 | 70063.531 | -0.002 | 0.999 |
| X14 | -540.239 | 107218.217 | -0.005 | 0.996 |
| X15 | 31.709 | 12943.540 | 0.002 | 0.998 |
| X16 | 73.749 | 13649.606 | 0.005 | 0.996 |
| X17 | 52.341 | 57193.234 | 0.001 | 0.999 |
| X18 | 90.037 | 20354.300 | 0.004 | 0.996 |
| X19 | -93.137 | 24992.864 | -0.004 | 0.997 |
| X20 | -262.422 | 65894.489 | -0.004 | 0.997 |
| X21 | -1030.876 | 195279.900 | -0.005 | 0.996 |
| X22 | 105.940 | 15834.642 | 0.007 | 0.995 |
| X23 | -502.589 | 175318.352 | -0.003 | 0.998 |
| X24 | 2156.153 | 279421.092 | 0.008 | 0.994 |
| X25 | -97.368 | 20531.897 | -0.005 | 0.996 |
| X26 | 5.471 | 30882.296 | 0.000 | 1.000 |
| X27 | 106.239 | 52714.248 | 0.002 | 0.998 |
| X28 | 162.495 | 25575.212 | 0.006 | 0.995 |
| X29 | 241.662 | 25846.588 | 0.009 | 0.993 |
| X30 | -39.356 | 62867.795 | -0.001 | 1.000 |

From the previous table, we can see that the relatively larger magnitude of X1 and X3 coefficients are similar but one is positive and one is negative. They also have relatively larger standard errors. It also fits the characteristics of the collinearity of the predictors we talked about during the class. Therefore, it leads to the instability of the coefficient estiamtes in the model and we overestimate the coefficients of these predictors.

## 1(f)

Table 2: Confusion Matrix of Training Set

| | Benign | Malignant |
|---|---|---|
| Benign | 255 | 0 |
| Malignant | 0 | 145 |

Table 3: Confusion Matrix of Test Set

| | Benign | Malignant |
|---|---|---|
| Benign | 99 | 7 |
| Malignant | 3 | 60 |

From the confusion matrix of the training set, we can see there is no missclassification error for whole training set, and the overall accurate rate is 94.083 for the testing set. However, none misclassification may help us noticing there is a problem of overfitting.

## Problem 2: Ridge Logistic Regression

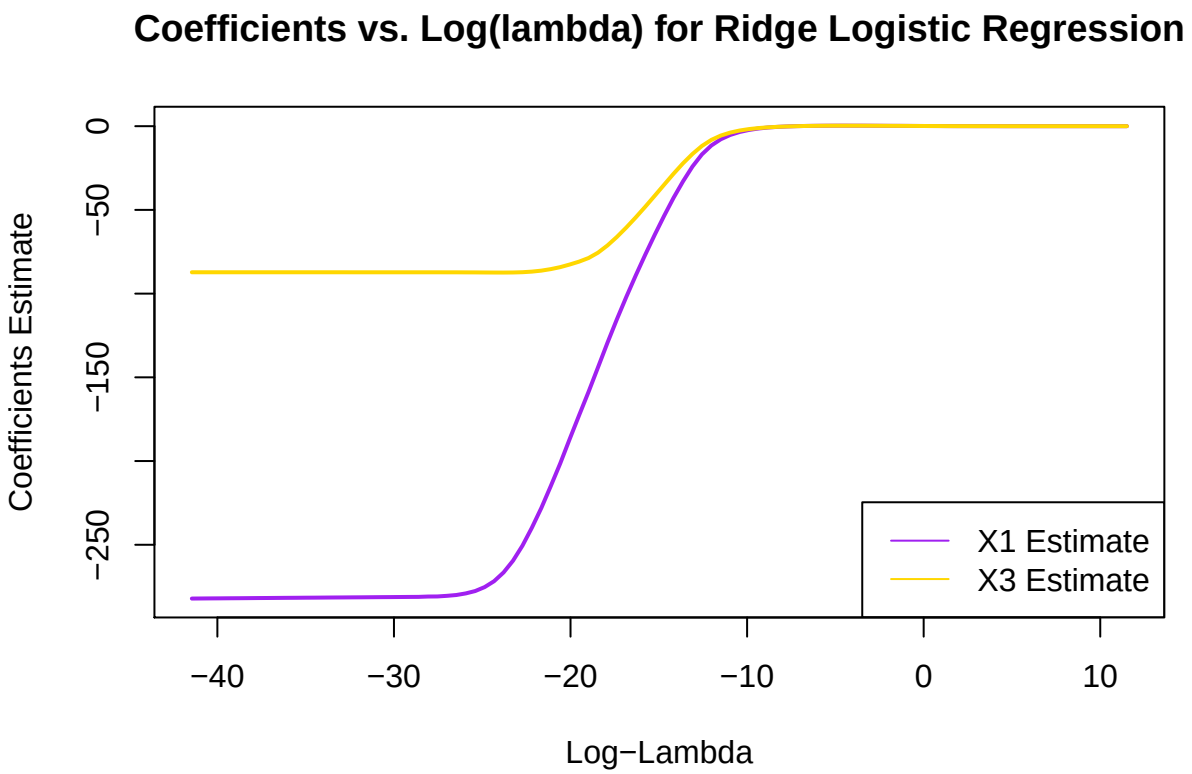
### 2(a)

The process will be shown in the appendix code section.

### 2(b)

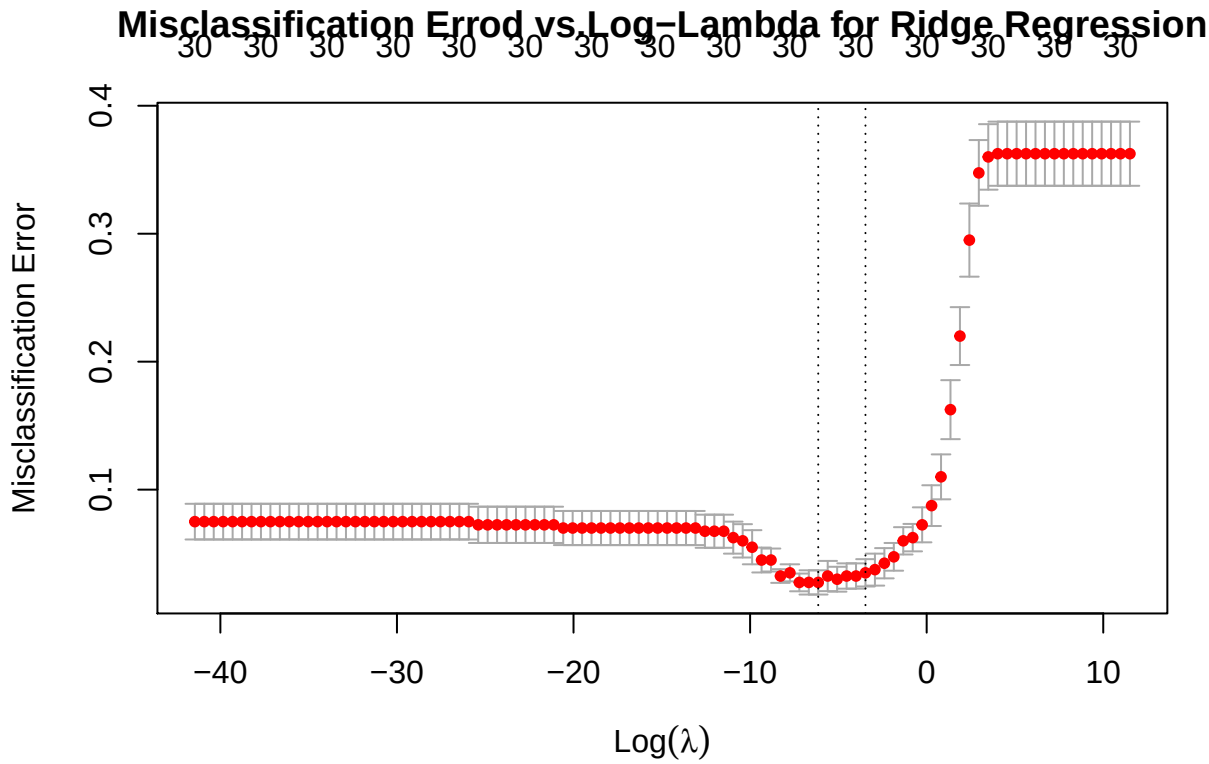
The process will be shown in the appendix code section.

### 2(c)




As log-lambda increase, both of the estimation are more towards to 0. The rate of X1 estimate going toward 0 is much higher than the X3 estimate. X1 estimate increase rapidly when log-lambda equals from -25 to -10. X3 estimate almost constatn when log-lambda smaller than -20, and increase after that. When Log-Lambda is around -10, both of the estimates are nearly equals to 0.

## 2(d)




The optimal lambda is 0.002, with log-lambda equals to -6.14.

## 2(e)

We estimate that the number of coefficients that are different from 0 for the ridge model is 30. It seemed to make sense since ridge regression only shrink all regression coeeficients toward to zero but will not give a set of zerio regression coefficients like LASSO does.

## 2(f)

Table 4: Confusion Matrix of Training Set

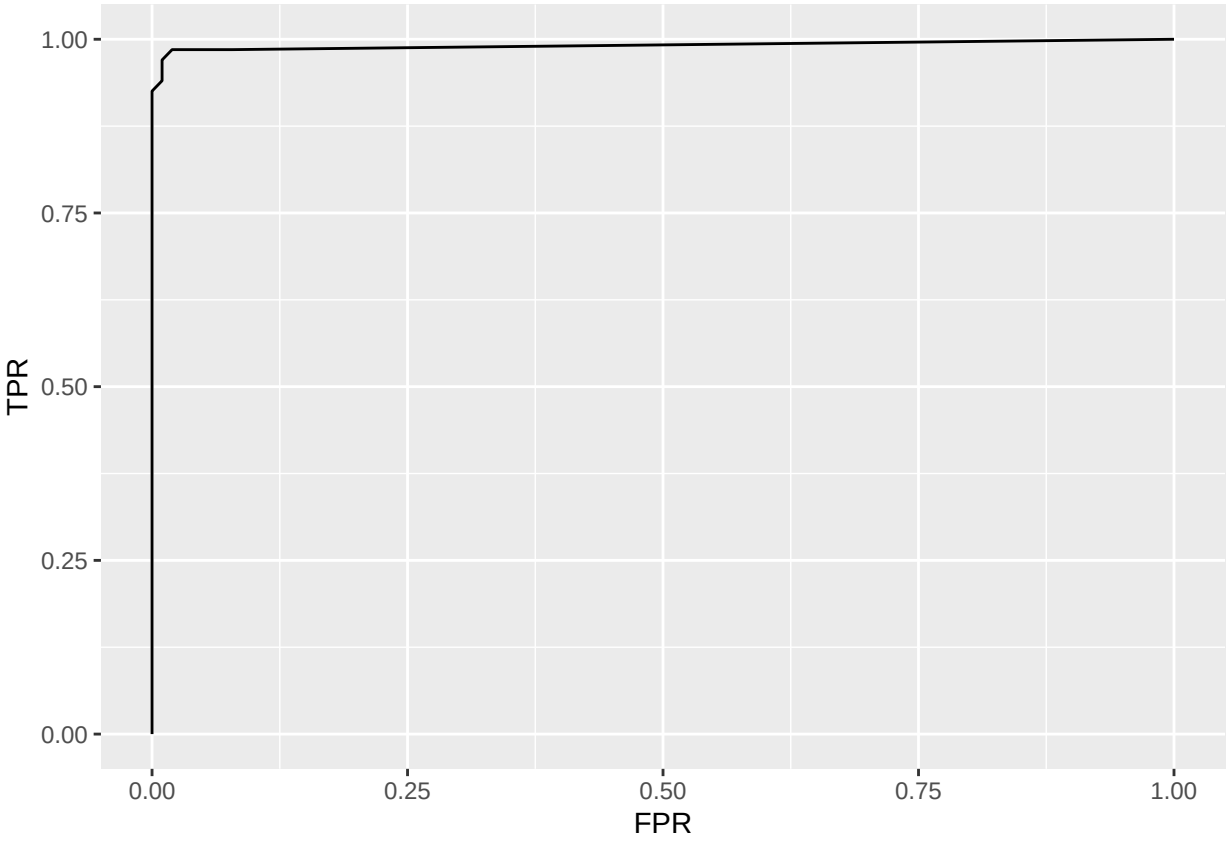
| | Benign | Malignant |
|---|---|---|
| Benign | 255 | 0 |
| Malignant | 0 | 145 |

Table 5: Confusion Matrix of Test Set

| | Benign | Malignant |
|---|---|---|
| Benign | 101 | 2 |
| Malignant | 1 | 65 |

We have a prediction accuracy of 100% in the training set and 98.225% prediction accuracy in the test set. Although 100% training set prediction accuracy seemed to be overfitting, the test set prediction accuracy is also incredibly high. Because we are randomly splitting the training set and the test set, and this model does well on both, we can say this model overall is giving very good prediction.

### 2(g)




### 2(h)

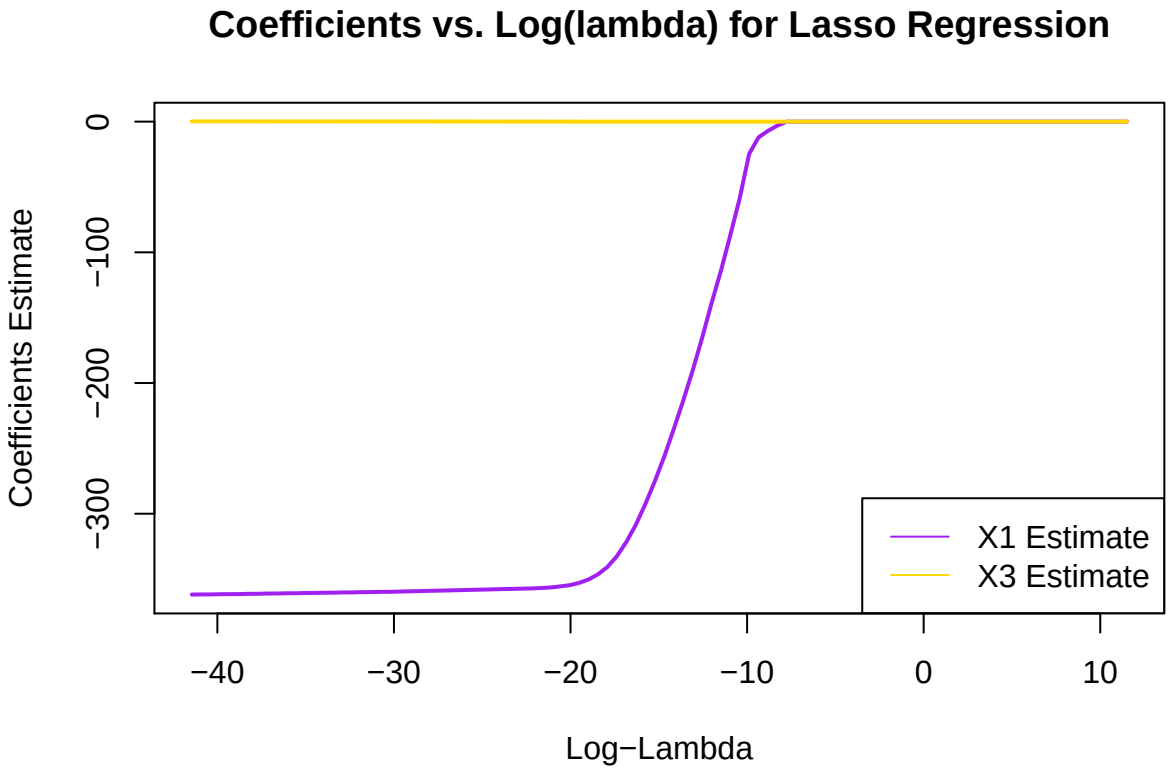
We calculated the AUC for ridge regression is 0.991‘.

## Problem 3: LASSO

### 3(b)

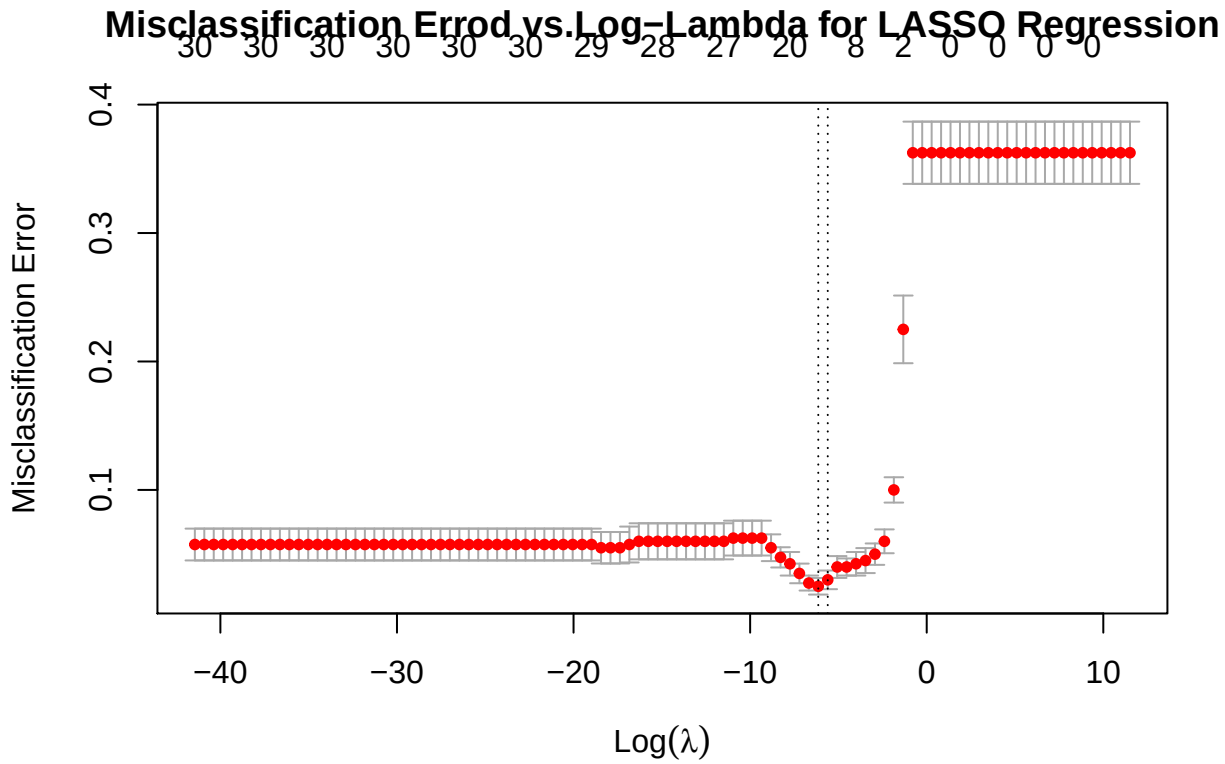
The process will be shown in the appendix code section.

**3(c)**




As log-lambda increase, the X1 estimate is more towards to 0. X1 estimate increase rapidly when log-lambda equals from -20 to -10, and equals to 0 when log-lambda is larger than (around)-8. X3 estimate is always zero for the whole time.

### 3(d)




The optimal lambda is 0.002, with log-lambda equals to -6.14.

### 3(e)

We estimate that the number of coefficients that are different from 0 for the LASSO model is 30. It seemed to make sense because LASSO regression will push the estimated coefficients to 0 when lambda becomes larger.

### 3(f)

Table 6: Confusion Matrix of Training Set

| | Benign | Malignant |
|---|---|---|
| Benign | 255 | 0 |
| Malignant | 0 | 145 |

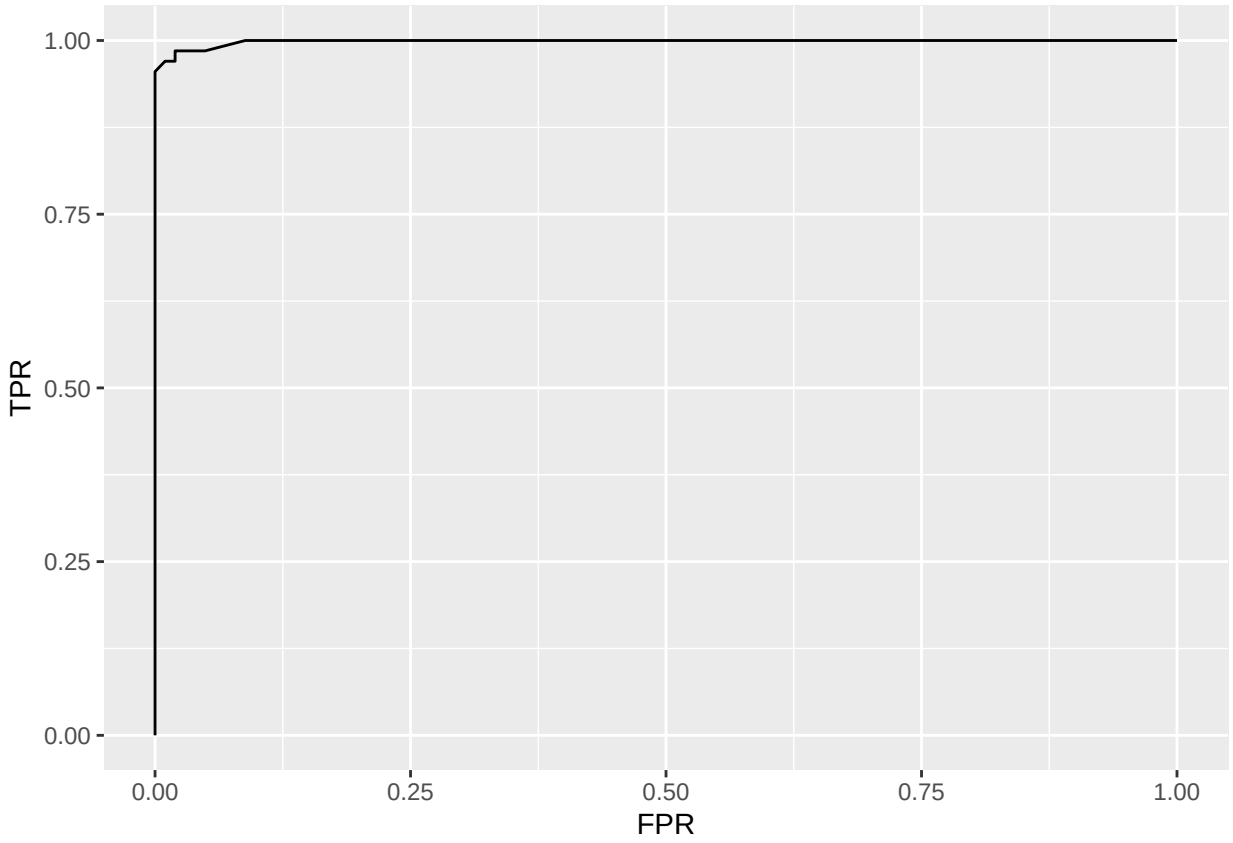
Table 7: Confusion Matrix of Test Set

| | Benign | Malignant |
|---|---|---|
| Benign | 101 | 2 |
| Malignant | 1 | 65 |

We have a prediction accuracy of 100% in the training set and 98.225% prediction accuracy in the test set. Although 100% training set prediction accuracy seemed to be overfitting, the test set prediction accuracy is

also incredibly high. Because we are randomly splitting the training set and the test set, and this model does well on both, we can say this model overall is giving very good prediction.

### 3(g)




### 3(h)

We calculated the AUC for LASSO regression is 0.999.

## Problem 4

All of these three models giving a 100% accuracy rate in the training set. However, for LASSO and ridge models, these models are giving better accuracy rate than the simple glm does. The auc of LASSO can reach to 0.999 which is very high and the auc of the ridge can reach to 0.991, showing these two models are doing better job in prediction compared to simple glm (with LASSO doing the best). In addition, LASSO only used 15 predictors but ridge used 30. Therefore, LASSO model is a simpler model with higher auc. The LASSO model is simpler also means that it is easier for us to interpret the LASSO model compared to the rest two models.

```
knitr::opts_chunk$set(echo = FALSE, warning = F, message = F)
#1(a)
library(readr)
library(dplyr)
library(ggplot2)
library(knitr)
wdbc <- read.csv("~/Desktop/R hw/wdbc.data", header=FALSE)[,-1]
colnames(wdbc)= c("diagnosis", paste0("X",1:30))
wdbc$diagnosis = factor(wdbc$diagnosis, levels = c("B","M"), labels = c("Benign","Malignant"))
n = nrow(wdbc)
n_pr = ncol(wdbc) - 1
n_m = wdbc%>%group_by(diagnosis)%>%summarize(n = n())

#1(b)
set.seed(2)
train_ids = sample(nrow(wdbc), 400)
test_ids = seq(nrow(wdbc))[-train_ids]
train_df = wdbc[train_ids,]
test_df = wdbc[test_ids,]
#1(c)
train.num = scale(train_df[,-1])
test.num = scale(test_df[,-1])
#1(d)
library(ggcorrplot)
library(ggplot2)
corr = cor(train.num)
ggcorrplot(corr)+ggtitle("Correlation Matrix of Training Predictors")
#1(e)
library(tidyr)
library(knitr)
train_df[,c(2:31)] = train.num
test_df[,c(2:31)] = test.num
glm.1 = glm(diagnosis~., data = train_df, family = binomial(link = "logit") )
kable(round(summary(glm.1)$coefficients,3),
      caption = "Estimated Coefficients of Simple Logistic Regression Using All Predictors")
corr_13 = summary(glm.1, correlation = TRUE)$correlation[2,4]
#1(f)
glm.prob.train = predict(glm.1, type = "response")
glm.label.train = rep("Benign", nrow(train_df))
glm.label.train[glm.prob.train > .5] = "Malignant"
tt.glm.train = table(True = glm.label.train, Predicted = train_df$diagnosis)
kable(tt.glm.train, caption = " Confusion Matrix of Training Set")
glm.prob.test = predict(glm.1,type = "response", newdata = test_df)
glm.label.test = rep("Benign", nrow(test_df))
glm.label.test[glm.prob.test > .5] = "Malignant"
tt.glm.test = table(glm.label.test, test_df$diagnosis)
kable(tt.glm.test, caption = " Confusion Matrix of Test Set")
accuracy_train = mean(glm.label.train == train_df$diagnosis)*100
accuracy_test = mean(glm.label.test == test_df$diagnosis)*100
#2(a)
X_train = as.matrix(train_df[,-1])
y_train = as.factor(train_df[,1])
X_test = as.matrix(test_df[,-1])
```

```
y_test = as.factor(test_df[,1])
#2(b)
library(glmnet)
grid = 10^seq(5,-18,length=100)
ridge.mod = glmnet(X_train, y_train, alpha=0,lambda = grid,thresh =1e-8, family = "binomial")
#2(c)
beta_13 = t(ridge.mod$beta[c("X1","X3"),])
beta_13 = cbind(beta_13, log(ridge.mod$lambda))
colnames(beta_13)[3] = "Log-Lambda"

plot(data = as.data.frame(as.matrix(beta_13)), X1~`Log-Lambda`, type = "l", col = "purple",
      ylab = "Coefficients Estimate",
     main = "Coefficients vs. Log(lambda) for Ridge Logistic Regression",
     lwd = 2)
lines(data = as.data.frame(as.matrix(beta_13)), X3~`Log-Lambda`, type = "l", col = "gold",lwd = 2)
legend("bottomright",legend = c("X1 Estimate","X3 Estimate"), col = c("purple","gold"),
       lty = 1)


#2(d)
cv.out = cv.glmnet(x = as.matrix(X_train), y = y_train, alpha=0,lambda = grid,thresh =1e-8,
                   family = "binomial",
                   type.measure = "class" )
plot(cv.out, main = "Misclassification Errod vs.Log-Lambda for Ridge Regression")
bestlam = cv.out$lambda.min

#2(e)
best_mod = glmnet(X_train,y_train,alpha =0, lambda = bestlam, family = "binomial",thresh =1e-8)
num_coef = sum(as.matrix(best_mod$beta)!=0)
#2(f)
ridge.prob.train = predict(best_mod, type = "response",newx = X_train)
ridge.label.train = rep("Benign", nrow(train_df))
ridge.label.train[ridge.prob.train > .5] = "Malignant"
tt.ridge.train = table(True = ridge.label.train, Predicted = y_train)
kable(tt.glm.train, caption = " Confusion Matrix of Training Set")
ridge.prob.test = predict(best_mod,type = "response", newx = X_test)
ridge.label.test = rep("Benign", nrow(X_test))
ridge.label.test[ridge.prob.test > .5] = "Malignant"
tt.ridge.test = table(ridge.label.test, y_test)
kable(tt.ridge.test, caption = " Confusion Matrix of Test Set")
accuracy_ridge = mean(ridge.label.test == y_test) *100
#2(g)
n_segm = 20
TPR = replicate(n_segm, 0)
FPR = replicate(n_segm, 0)
p_th = seq(0,1,length.out = n_segm)
for (i in 1:n_segm)
{
  ridge.label.test = rep("Benign", nrow(test_df))
  ridge.label.test[ridge.prob.test > p_th[i]] = "Malignant"

  tt.ridge.test = table(ridge.label.test, test_df$diagnosis)
  TPR[i] = mean(ridge.label.test[test_df$diagnosis == 'Malignant'] == test_df$diagnosis[test_df$diagnos
```

```
  FPR[i] = mean(ridge.label.test[test_df$diagnosis == 'Benign'] != test_df$diagnosis[test_df$diagnosis =
}
ggplot() + geom_path(aes(x = FPR, y = TPR))


#2(h)
library(pracma)
auc = abs(trapz(FPR, TPR))
#3(b)
lasso.mod = glmnet(X_train, y_train, alpha=1,lambda = grid,thresh =1e-8, family = "binomial")
#3(c)
beta_13 = t(lasso.mod$beta[c("X1","X3"),])
beta_13 = cbind(beta_13, log(lasso.mod$lambda))
colnames(beta_13)[3] = "Log-Lambda"

plot(data = as.data.frame(as.matrix(beta_13)), X1~`Log-Lambda`, type = "l", col = "purple",
      ylab = "Coefficients Estimate",
     main = "Coefficients vs. Log(lambda) for Lasso Regression",
     lwd = 2)
lines(data = as.data.frame(as.matrix(beta_13)), X3~`Log-Lambda`, type = "l", col = "gold",lwd = 2)
legend("bottomright",legend = c("X1 Estimate","X3 Estimate"), col = c("purple","gold"),
       lty = 1)
#3(d)
cv.out.lasso = cv.glmnet(x = as.matrix(X_train), y = y_train, alpha=1,lambda = grid,thresh =1e-8,
                   family = "binomial",
                   type.measure = "class" )
plot(cv.out.lasso, main = "Misclassification Errod vs.Log-Lambda for LASSO Regression")
bestlam.lasso = cv.out.lasso$lambda.min

#3(e)
best_mod_lasso = glmnet(X_train,y_train,alpha =1, lambda = bestlam.lasso, family = "binomial",thresh =1
num_coef = sum(as.matrix(best_mod$beta)!=0)
#3(f)
lasso.prob.train = predict(best_mod_lasso, type = "response",newx = X_train)
lasso.label.train = rep("Benign", nrow(train_df))
lasso.label.train[lasso.prob.train > .5] = "Malignant"
tt.lasso.train = table(True = lasso.label.train, Predicted = y_train)
kable(tt.glm.train, caption = " Confusion Matrix of Training Set")
lasso.prob.test = predict(best_mod_lasso,type = "response", newx = X_test)
lasso.label.test = rep("Benign", nrow(X_test))
lasso.label.test[lasso.prob.test > .5] = "Malignant"
tt.lasso.test = table(lasso.label.test, y_test)
kable(tt.lasso.test, caption = " Confusion Matrix of Test Set")
accuracy_lasso = mean(lasso.label.test == y_test) *100
#2(g)
n_segm = 20
TPR = replicate(n_segm, 0)
FPR = replicate(n_segm, 0)
p_th = seq(0,1,length.out = n_segm)
for (i in 1:n_segm)
{
  lasso.label.test = rep("Benign", nrow(test_df))
  lasso.label.test[lasso.prob.test > p_th[i]] = "Malignant"
```

```
  tt.lasso.test = table(lasso.label.test, test_df$diagnosis)
  TPR[i] = mean(lasso.label.test[test_df$diagnosis == 'Malignant'] == test_df$diagnosis[test_df$diagnosi
  FPR[i] = mean(lasso.label.test[test_df$diagnosis == 'Benign'] != test_df$diagnosis[test_df$diagnosis =
}
ggplot() + geom_path(aes(x = FPR, y = TPR))


#3(h)
auc = abs(trapz(FPR, TPR))
```