STAT 6340 Statistical Machine Learning

Mini Project 4

Author:
Suarez Rodes, Randy

Supervisor:
Choudhary, Pankaj Ph.D.

March 31, 2021

1 Answers

Question 1.a We fitted a model using all predictors obtaining an estimated test mean squared error of 1.135158.

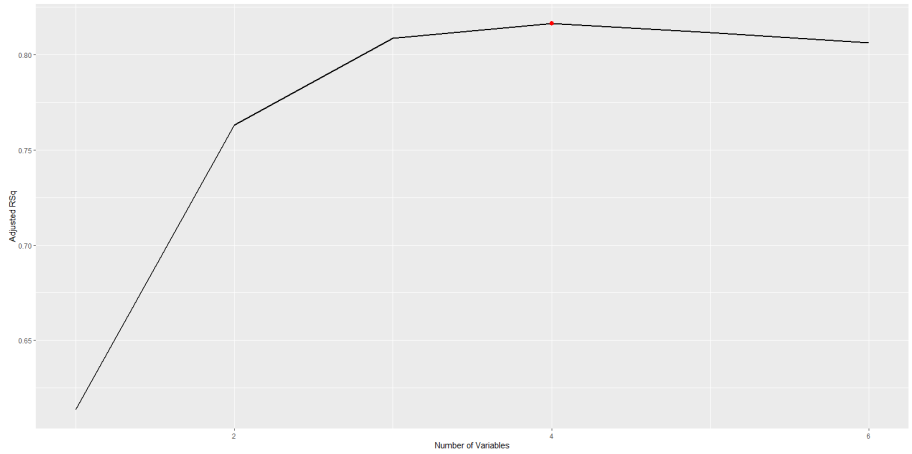

Figure 1: Best-subset, Forward and Backward Stepwise Model (4 variables)

Question 1.b, c and d Based on the adjusted R^2 we obtained a model with 4 variables Figure 1. The same model was obtained from the best-subset, forward and backward stepwise selection methods. This model have the predictors Flavor, Oakiness and Region (categorical variable of three levels), meaning we ended up with a model of 4 variables by encoding Region into two variables with one base value. The estimated test mean squared error is 0.8705717.

Question 1.e We run LOOCV to find the best lambda parameter for Ridge Regression. We used a grid of 100 possible lambdas ranging from 10^{-3} to 10^{10}. The results of this process can be found on Figure 2. The minimal test MSE = 1.083897 was obtained for lambda = 0.3125716 (represented along the coefficients on Figure 3).

Question 1.f We run LOOCV to find the best lambda parameter for Lasso. We used the same grid as in question 1.e. The results of this process can be found on Figure 4. The minimal test MSE = 0.9990885 was obtained for lambda = 0.1261857 (represented along the coefficients on Figure 5). This method actually provides us variable selection by making the predictors Clarity and Body equal zero.

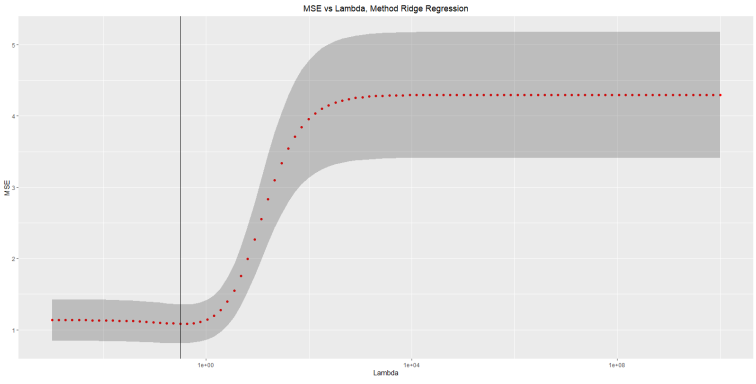

Figure 2: MSE vs Lambda, Ridge Reg

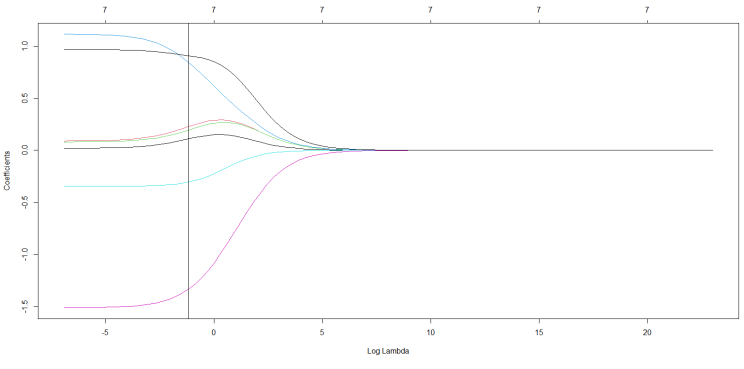

Figure 3: Ridge Coefficients

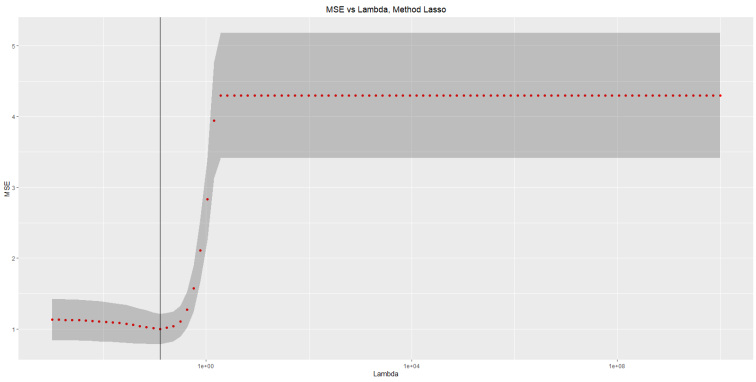

Figure 4: MSE vs Lambda, Lasso

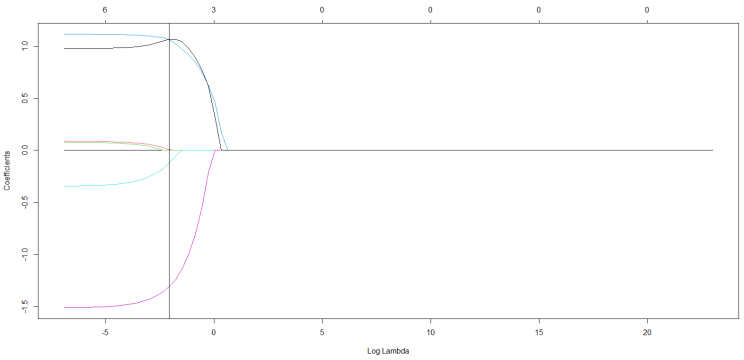

Figure 5: Lasso Coefficients

Question 1.g We present on Table 1 a summary of the estimated test MSE along with coefficient estimates for each model.

	Full	Subset,Forward & Backward	RidgeReg	Lasso
MSE	1.1352	0.8706	1.0839	0.9991
Intercept	7.8144	8.1208	7.6040	7.8388
Clarity	0.0171	-	0.1089	0.0000
Aroma	0.0890	-	0.2293	0.0045
Body	0.0797	-	0.1951	0.0000
Flavor	1.1172	1.1920	0.8456	1.0631
Oakiness	-0.3464	-0.3183	-0.3047	-0.1224
Region2	-1.5129	-1.5155	-1.3345	-1.3107
Region3	0.9726	1.0935	0.9096	1.0708

Table 1: Summary of estimated Test MSE and Coefficients

We recommend the model resulting from the Best-Subset selection method which offers the best MSE and the lowest complexity of all models.

Question 2.a We fitted a logistic regression model using all predictors obtaining an estimated test error rate of 22.198% from a 10 folds cross validation approach.

Question 2.b, c and d Based on AIC the best model obtained has 7 predictors. The same model was obtained from the best-subset, forward and backward stepwise selection methods. All predictors are part of the model except for SkinThickness. The estimated test error rate obtained from cross validation is 22.098%.

Question 2.e Using the same lambda grid as in experiment 1 and a 10-fold cross validation method to estimate the test error rate, we find the best lambda parameter for Ridge Regression. The results of this process can be found on Figure 6. The minimum test error rate is 22% and it was obtained for lambda = 0.004534879 (represented along the coefficients on Figure 7).

Question 2.f With the same lambda grid and 10-fold cross validation method to estimate the test error rate we look for the best value of lambda. The results of this process can be found on Figure 8. The minimum test error rate is 22.10% and it was obtained for lambda = 0.001 (represented along the coefficients on Figure 9). Lasso also drop the predictor SkinThickness making it equal to zero.

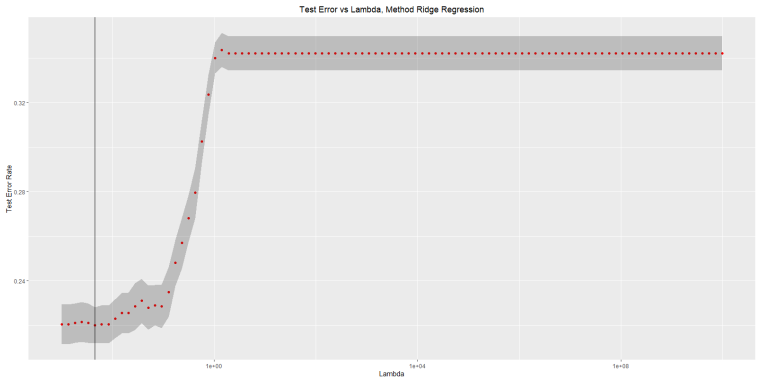

Figure 6: Error Rate vs Lambda, Ridge Reg

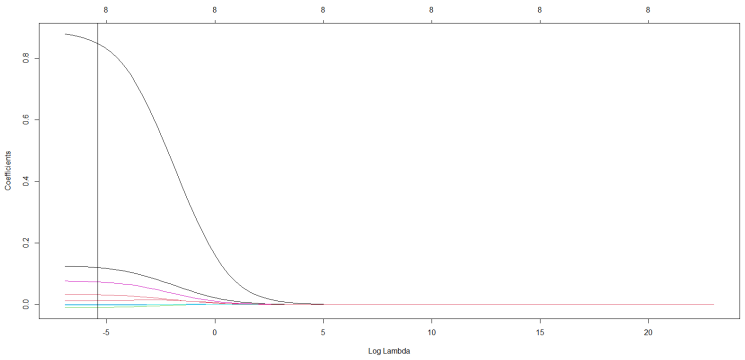

Figure 7: Ridge Coefficients

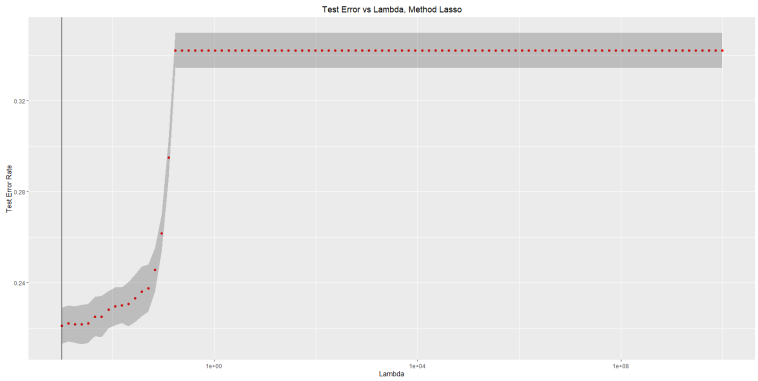

Figure 8: Error Rate vs Lambda, Lasso

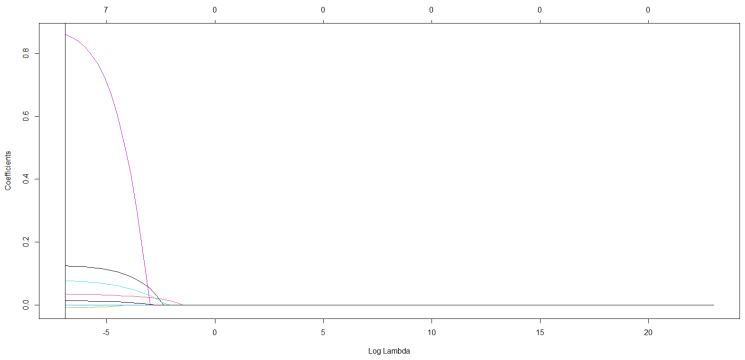

Figure 9: Lasso Coefficients

Question 2.g We present on Table 2 a summary of the estimated test errors along with coefficient estimates for each model.

	Full	Subset, Forward & Backward	RidgeReg	Lasso
Error	0.22198	0.22098	0.22000	0.22100
Intercept	-8.02645	-8.02731	-7.76769	-7.93907
Pregnancies	0.12638	0.12637	0.12034	0.12417
Glucose	0.03372	0.03368	0.03221	0.03329
BloodPressure	-0.00964	-0.00958	-0.00885	-0.00898
SkinThickness	0.00052	-	0.00041	0.00000
Insulin	-0.00124	-0.00121	-0.00107	-0.00112
BMI	0.07755	0.07787	0.07407	0.07616
DiabetesPedigreeFunction	0.88776	0.88949	0.84856	0.86156
Age	0.01294	0.01289	0.01363	0.01262

Table 2: Summary of estimated Test Error and Coefficients

All models are very close in accuracy. Ridge Regression slightly has better error than the other models, although this model includes the predictor SkinThickness. Previous analysis from project 3 tell us that this predictor is not relevant to predict the presence of diabetes, therefore including this predictor might lead to overfitting. Due to this fact we recommend the use of the best-subset selection model since it does not include this problematic predictor and it has a slight less error rate than Lasso, hence we ended up with the same model as in project 3.

2 Code

```
#Packages
library(ggplot2) #Used for graphics and visual representations

library(MASS) #Used for LD and QD analysis
library(caret) #Used to handle LOOCV training

library(leaps) #Used for best-subset and forward and backward stepwise selection
               #with linear models
library(bestglm) #Used for best-subset and forward and backward stepwise selection
                 #with generalized linear models
library(glmnet) #Use for Ridge Regression and Lasso
library(broom) #To create tidy objects for ggplot visualization

rdseed=8466 #Seed to replicate results

#Experiment 1
print("Experiment 1")

#Reading the data
wine=read.delim("wine.txt")

#Converting Region to factor
wine$Region=as.factor(wine$Region)

#Dataframe to track errors of each model
error.df=data.frame(Full=0,Subset=0,Forward=0,Backward=0,RidgeReg=0,Lasso=0)
row.names(error.df)="MSE"

#Dataframe to track coefficients estimates
df.coeff.estimates=as.data.frame(matrix("",nrow=8,ncol = 6))

names(df.coeff.estimates)=c("Full","Subset","Forward","Backward","RidgeReg","Lasso")

#Defining the training control for the train method of caret
control=trainControl(method = "LOOCV")

#Number of observations
n=nrow(wine)


####Question 1.a####

#LOOCV on full regression model
m_fullloocv = train(Quality ~ .,
              data = wine,
              method = "lm",
              trControl = control)

print(m_fullloocv)

#Estimated MSE
error.df$Full=m_fullloocv$results$RMSE^2

#Estimated coefficients
mfull=lm(Quality~.,data=wine)
coeff.full=mfull$coefficients

row.names(df.coeff.estimates)=names(coeff.full)
df.coeff.estimates[,1]=coeff.full
```

```
####Question 1.b####

#Total predictors
totpred=ncol(wine)-1

#Finding best-subset selection models
m.subset=regsubsets(Quality ~ ., wine, nvmax = totpred)
m.subset.summ=summary(m.subset)

#Best model according to adjusted r2
kbest=which.max(m.subset.summ$adjr2)

#Visualizing adj r2 vs number of predictors
print(ggplot(data.frame(predictors = 1:totpred, adj_R2 = m.subset.summ$adjr2),
       aes(x=predictors,y=adj_R2))+geom_line(size=1)+
  geom_point(aes(x=kbest,y=m.subset.summ$adjr2[kbest]),
             color="red",size=4,shape=20)+
  labs(x="Number of Variables",y="Adjusted RSq")+
  theme(plot.title=element_text(hjust=0.5))+
  ggtitle("Best Subset Selection Adjusted RSq"))

#Exploring the coefficients of the best model
selected_predictors=coef(m.subset, kbest)
print(selected_predictors) #Clarity, Aroma and Body were dropped

#LOOCV of the best model
msub_LOOCV= train(Quality ~. -Clarity -Aroma -Body,
                  data = wine,
                  method = "lm",
                  trControl = control)

print(msub_LOOCV)

#Estimated MSE
error.df$Subset=msub_LOOCV$results$RMSE^2

#Final model
m.subset=lm(Quality ~. -Clarity -Aroma -Body,data=wine)
#Estimated coefficients
coeff.subset=m.subset$coefficients

df.coeff.estimates[-(2:4),2]=coeff.subset


####Question 1.c####

#Finding forward selection models
m.forward=regsubsets(Quality ~ ., wine, nvmax = totpred,method="forward")
m.forward.summ=summary(m.forward)

#Best model according to adjusted r2
k.forward=which.max(m.forward.summ$adjr2)

#Visualizing adj r2 vs number of predictors
print(ggplot(data.frame(predictors = 1:totpred, adj_R2 = m.forward.summ$adjr2),
       aes(x=predictors,y=adj_R2))+geom_line(size=1)+
  geom_point(aes(x=k.forward,y=m.forward.summ$adjr2[k.forward]),
             color="red",size=4,shape=20)+
  labs(x="Number of Variables",y="Adjusted RSq")+
  theme(plot.title=element_text(hjust=0.5))+
  ggtitle("Forward Stepwise Selection Adjusted RSq"))
```

```r
#Exploring the coefficients of the best model
selected_predictors=coef(m.forward, k.forward)
print(selected_predictors)
#Same predictors as best-subset selection,
#therefore we will end with the same Estimated MSE and coefficients

#Estimated MSE
error.df$Forward=error.df$Subset

#Estimated coefficients same as best subset selection
df.coeff.estimates[-(2:4),3]=coeff.subset


####Question 1.d####

#Finding backward selection models
m.backward=regsubsets(Quality ~ ., wine, nvmax = totpred,method="backward")
m.backward.summ=summary(m.backward)

#Best model according to adjusted r2
k.backward=which.max(m.backward.summ$adjr2)

#Visualizing adj r2 vs number of predictors
print(ggplot(data.frame(predictors = 1:totpred, adj_R2 = m.backward.summ$adjr2),
       aes(x=predictors,y=adj_R2))+geom_line(size=1)+
  geom_point(aes(x=k.backward,y=m.backward.summ$adjr2[k.backward]),
             color="red",size=4,shape=20)+
  labs(x="Number of Variables",y="Adjusted RSq")+
  theme(plot.title=element_text(hjust=0.5))+
  ggtitle("Backward Stepwise Selection Adjusted RSq"))

#Exploring the coefficients of the best model
selected_predictors=coef(m.backward, k.backward)
print(selected_predictors)
#Same predictors as best-subset selection
#therefore we will end with the same Estimated MSE and coefficients

#Estimated MSE
error.df$Backward=error.df$Subset

#Estimated coefficients same as best subset selection
df.coeff.estimates[-(2:4),4]=coeff.subset

####Question 1.e####

#Defining predictors, response and lambdas
response=wine[,6]
predictors=model.matrix(Quality ~ ., wine)[, -1]
lambdas = 10^seq(10, -3, length = 100)

#Fitting ridge regression models for the different lambdas
m.ridge = glmnet(predictors, response, alpha = 0, lambda = lambdas)

#Applying LOOCV to find best lambda
cv.ridge = cv.glmnet(predictors, response, alpha = 0, nfolds = n,
                     lambda = lambdas, grouped = FALSE, type.measure = "mse")


#Tidy data frames to graph
tidy_cv <- tidy(cv.ridge)
glance_cv <- glance(cv.ridge)

#Plot of MSE as a function of lambda
```

```
g.ridge = ggplot(tidy_cv, aes(lambda, estimate))+
  geom_point(color="red") + scale_x_log10()+
  geom_ribbon(aes(ymin = conf.low, ymax = conf.high), alpha = .25)+
  geom_vline(xintercept = glance_cv$lambda.min)+
  labs(x="Lambda",y="MSE")+
  theme(plot.title=element_text(hjust=0.5))+
  ggtitle("MSE vs Lambda, Method Ridge Regression")

print(g.ridge)

#Best lambda value: 0.3125716
bestlamb_ridge=cv.ridge$lambda.min

#Visualizing best lambda and coefficients
plot(m.ridge, xvar = "lambda")
abline(v = log(bestlamb_ridge))

#Estimated MSE
error.df$RidgeReg=min(cv.ridge$cvm)

#Final model
m.ridge_final=glmnet(predictors, response, alpha = 0, lambda = bestlamb_ridge)
#Estimated coefficients
coeff.ridge=predict(m.ridge, type = "coefficients", s = bestlamb_ridge)[1:8, ]

df.coeff.estimates[,5]=coeff.ridge

####Question 1.f####

#Fitting lasso models for the different lambdas
m.lasso = glmnet(predictors, response, alpha = 1, lambda = lambdas)

#Applying LOOCV to find best lambda
cv.lasso = cv.glmnet(predictors, response, alpha = 1, nfolds = n,
                     lambda = lambdas, grouped = FALSE, type.measure = "mse")

#Tidy data frames to graph
tidy_cv <- tidy(cv.lasso)
glance_cv <- glance(cv.lasso)

#Plot of MSE as a function of lambda
g.lasso = ggplot(tidy_cv, aes(lambda, estimate))+
  geom_point(color="red") + scale_x_log10()+
  geom_ribbon(aes(ymin = conf.low, ymax = conf.high), alpha = .25)+
  geom_vline(xintercept = glance_cv$lambda.min)+
  labs(x="Lambda",y="MSE")+
  theme(plot.title=element_text(hjust=0.5))+
  ggtitle("MSE vs Lambda, Method Lasso")

print(g.lasso)

#Best lambda value: 0.1261857
bestlamb_lasso=cv.lasso$lambda.min

#Visualizing best lambda and coefficients
plot(m.lasso, xvar = "lambda")
abline(v = log(bestlamb_lasso))

#Estimated MSE
error.df$Lasso=min(cv.lasso$cvm)

#Final model
m.lasso_final=glmnet(predictors, response, alpha = 1, lambda = bestlamb_lasso)
```

```r
#Estimated coefficients
coeff.lasso=predict(m.lasso_final, type = "coefficients", s = bestlamb_lasso)[1:8, ]

df.coeff.estimates[,6]=coeff.lasso

####Question 1.g####

final.df=rbind(error.df,df.coeff.estimates)

print(final.df)

#Experiment 2
print("Experiment 2")

#Reading the data
diabetes = read.csv("diabetes.csv")

#Renaming predictors
names(diabetes)=c("Pregnancies", "Glucose", "BloodPressure", "SkinThickness",
                  "Insulin",   "BMI",
                  "DiabetesPedigreeFunction", "Age","Outcome" )

#Converting Outcome to factor
diabetes$Outcome=as.factor(diabetes$Outcome)

#Defining the training control for the train method of caret
#10 folds cross validation
set.seed(rdseed)
train_control=trainControl(method = "cv", number = 10)

#Dataframe to track errors of each model
error.df2=data.frame(Full=0,Subset=0,Forward=0,Backward=0,RidgeReg=0,Lasso=0)
row.names(error.df2)="Error"

#Dataframe to track coefficients estimates
df.coeff.estimates2=as.data.frame(matrix("",nrow=9,ncol = 6))

names(df.coeff.estimates2)=c("Full","Subset","Forward","Backward","RidgeReg","Lasso")

####Question 2.a####

#10 fold cv on full model of logistic regression
set.seed(rdseed)
m.full.cv = train(
  form = Outcome ~ .,
  data = diabetes,
  trControl = train_control,
  method = "glm",
  family = "binomial"
)

#Estimated test error rate
error.df2$Full=1-m.full.cv$results$Accuracy

#Estimated coefficients
m.full_glm=glm(Outcome~.,data = diabetes,family = binomial)

row.names(df.coeff.estimates2)=names(m.full_glm$coefficients)
df.coeff.estimates2[,1]=m.full_glm$coefficients
```

```
####Question 2.b####

#Redifining diabetes data frame to be used on bestglm package
diabetes_glm=diabetes
diabetes_glm$y=diabetes_glm$Outcome
diabetes_glm$Outcome=NULL

#Best-subset selection model according to AIC
fit.subset.logistic =  bestglm(Xy = diabetes_glm, family = binomial, IC = "AIC",
                               method = "exhaustive")

print(fit.subset.logistic$BestModel$coefficients) #SkinThickness was dropped


#10 fold cv on best subset model
set.seed(rdseed)
m.subset.cv = train(
  form = Outcome ~ .-SkinThickness,
  data = diabetes,
  trControl = train_control,
  method = "glm",
  family = "binomial"
)

#Estimated test error rate
error.df2$Subset=1-m.subset.cv$results$Accuracy

#Best model
m_log.subset=glm(Outcome ~ .-SkinThickness,data=diabetes,family = binomial)
#Estimated coefficients
c.subset=m_log.subset$coefficients
df.coeff.estimates2[-5,2]=c.subset


####Question 2.c####

fit.best.forward =  bestglm(Xy = diabetes_glm, family = binomial, IC = "AIC",
                               method = "forward")

print(fit.best.forward$BestModel$coefficients) #SkinThickness was dropped
#Same model as in best-subset selection method,
#therefore we will end with the same Estimated test error and coefficients

#Estimated test error rate
error.df2$Forward=error.df2$Subset

#Estimated coefficients
df.coeff.estimates2[-5,3]=c.subset


####Question 2.d####

fit.best.backward =  bestglm(Xy = diabetes_glm, family = binomial, IC = "AIC",
                              method = "backward")

print(fit.best.backward$BestModel$coefficients) #SkinThickness was dropped
#Same model as best-subset selection method
#therefore we will end with the same Estimated test error and coefficients

#Estimated test error rate
error.df2$Backward=error.df2$Subset

#Estimated coefficients
```

```
df.coeff.estimates2[-5,4]=c.subset


####Question 2.e####

#Defining predictors, response and lambdas
response_diab=diabetes[,9]
predictors_diab=model.matrix(Outcome ~ ., diabetes)[, -1]
lambdas = 10^seq(10, -3, length = 100)


#Fitting ridge regression models for the different lambdas
mlog.ridge = glmnet(predictors_diab, response_diab, alpha = 0, lambda = lambdas,
                    family = "binomial")


#Applying 10-fold cv to find best lambda
set.seed(rdseed)
cv10.ridge = cv.glmnet(predictors_diab, response_diab, alpha = 0, nfolds = 10,
                       lambda = lambdas, type.measure = "class", family = "binomial")

#Tidy data frames to graph
tidy_cv <- tidy(cv10.ridge)
glance_cv <- glance(cv10.ridge)

#Plot of MSE as a function of lambda
g.ridgecv = ggplot(tidy_cv, aes(lambda, estimate))+
  geom_point(color="red") + scale_x_log10()+
  geom_ribbon(aes(ymin = conf.low, ymax = conf.high), alpha = .25)+
  geom_vline(xintercept = glance_cv$lambda.min)+
  labs(x="Lambda",y="Test Error Rate")+
  theme(plot.title=element_text(hjust=0.5))+
  ggtitle("Test Error vs Lambda, Method Ridge Regression")

print(g.ridgecv)

#Lambda for the minimal test error rate
loglamb_ridge=cv10.ridge$lambda.min

#Visualizing best lambda and coefficients
plot(mlog.ridge, xvar = "lambda")
abline(v = log(loglamb_ridge))

#Estimated test error
error.df2$RidgeReg=min(cv10.ridge$cvm)


#Best model
m_log.ridge=glmnet(predictors_diab, response_diab, alpha = 0, lambda = loglamb_ridge,
                   family = "binomial")
#Estimated coefficients
c.ridge= predict(m_log.ridge, type = "coefficients", s = loglamb_ridge)[1:9, ]
df.coeff.estimates2[,5]=c.ridge


####Question 2.f####


#Fitting lasso models for the different lambdas
mlog.lasso = glmnet(predictors_diab, response_diab, alpha = 1, lambda = lambdas,
                    family = "binomial")

#Applying 10-fold cv to find best lambda
```

```
set.seed(rdseed)
cv10.lasso = cv.glmnet(predictors_diab, response_diab, alpha = 1, nfolds = 10,
                       lambda = lambdas, type.measure = "class", family = "binomial")

#Tidy data frames to graph
tidy_cv <- tidy(cv10.lasso)
glance_cv <- glance(cv10.lasso)

#Plot of MSE as a function of lambda
g.lassocv = ggplot(tidy_cv, aes(lambda, estimate))+
  geom_point(color="red") + scale_x_log10()+
  geom_ribbon(aes(ymin = conf.low, ymax = conf.high), alpha = .25)+
  geom_vline(xintercept = glance_cv$lambda.min)+
  labs(x="Lambda",y="Test Error Rate")+
  theme(plot.title=element_text(hjust=0.5))+
  ggtitle("Test Error vs Lambda, Method Lasso")

print(g.lassocv)


#Estimated test error rate
error.df2$Lasso=min(cv10.lasso$cvm)


#Lambda for the minimal test error rate
loglamb_lasso=cv10.lasso$lambda.min

#Visualizing best lambda and coefficients
plot(mlog.lasso, xvar = "lambda")
abline(v = log(loglamb_lasso))

#Best model
m_log.lasso=glmnet(predictors_diab, response_diab, alpha = 1, lambda = loglamb_lasso,
                   family = "binomial")
#Estimated coefficients
c.lasso = predict(m_log.lasso, type = "coefficients", s = loglamb_lasso)[1:9, ]
df.coeff.estimates2[,6]=c.lasso



####Question 2.g####

final.df2=rbind(error.df2,df.coeff.estimates2)

print(final.df2)
```