```
#set working directory if needed (modify path as needed)
#setwd("C:/Users/Kailash/Documents/logistic")
#load required library
library(mlbench)
#load Pima Indian Diabetes dataset
data("PimaIndiansDiabetes")
#set seed to ensure reproducible results
set.seed(42)
#split into training and test sets
PimaIndiansDiabetes[,"train"] <-
ifelse(runif(nrow(PimaIndiansDiabetes))<0.8,1,0)
#separate training and test sets
trainset <- PimaIndiansDiabetes[PimaIndiansDiabetes$train==1,]
testset <- PimaIndiansDiabetes[PimaIndiansDiabetes$train==0,]
#get column index of train flag
trainColNum <- grep("train",names(trainset))
#remove train flag column from train and test sets
trainset <- trainset[,-trainColNum]
testset <- testset[,-trainColNum]
#get column index of predicted variable in dataset
typeColNum <- grep("diabetes",names(PimaIndiansDiabetes))
#build model
glm_model <- glm(diabetes~.,data = trainset, family = binomial)
summary(glm_model)

#predict probabilities on testset
#type="response" gives probabilities, type="class" gives class
glm_prob <- predict.glm(glm_model,testset[,-typeColNum],type="response")
#which classes do these probabilities refer to? What are 1 and 0?
contrasts(PimaIndiansDiabetes$diabetes)

#make predictions
##…first create vector to hold predictions (we know 0 refers to neg now)
glm_predict <- rep("neg",nrow(testset))
glm_predict[glm_prob>.5] <- "pos"
#confusion matrix
table(pred=glm_predict,true=testset$diabetes)

#accuracy
mean(glm_predict==testset$diabetes)

#load required library
library(glmnet)
#convert training data to matrix format
x <- model.matrix(diabetes~.,trainset)
#convert class to numerical variable
y <- ifelse(trainset$diabetes=="pos",1,0)
#perform grid search to find optimal value of lambda
#family= binomial => logistic regression, alpha=1 => lasso
#error measure is important, we choose mse, but check others in documentation
cv.out <- cv.glmnet(x,y,alpha=1,family="binomial",type.measure = "mse" )
#plot result
```

```
plot(cv.out)


#min value of lambda
min_lambda <- cv.out$lambda.min
#regression coefficients
coef(cv.out)

#get test data
x_test <- model.matrix(diabetes~.,testset)
#predict class, type="class"
lasso_prob <- predict.glm(glm_model,testset[,-typeColNum],type="response")
#translate probabilities to predictions
lasso_predict <- rep("neg",nrow(testset))
lasso_predict[lasso_prob>.5] <- "pos"
#confusion matrix
table(pred=lasso_predict,true=testset$diabetes)

#accuracy
mean(lasso_predict==testset$diabetes)
```