Project 3, STAT 557

Balaji Kumar, bzk18

4/17/2017

```
#Reading in the data
rm(list=ls())
setwd("/data/bzk18/Project/Misc/Spring17/STAT557/Proj3/")
t1 = Sys.time()
X = read.table("datX.txt",sep=",")
Y = (read.table("labels.txt",sep="\n"))
Y = Y[,1]
dim(X)
```

```
## [1] 20000  1024
```

```
X = X[!duplicated(as.list(X))]
dim(X)
```

```
## [1] 20000   512
```

```
X = scale(log(X+0.1),center=T,scale=T)
X = data.frame(X)
t2 = Sys.time()
print(t2-t1)
```

```
## Time difference of 7.483396 secs
```

```
#Dividing training and test sets
t1 = Sys.time()
n = nrow(X)
Y = as.factor(Y)
set.seed(100)
train = sample(1:n, round(3*n/4))
X.train = X[train,]
Y.train = as.factor(Y[train])
X.test = X[-train,]
Y.test = as.factor(Y[-train])
data.train = cbind(X.train,Y.train)
data.test = cbind(X.test, Y.test)
t2 = Sys.time()
print(t2-t1)
```

```
## Time difference of 0.2362394 secs
```

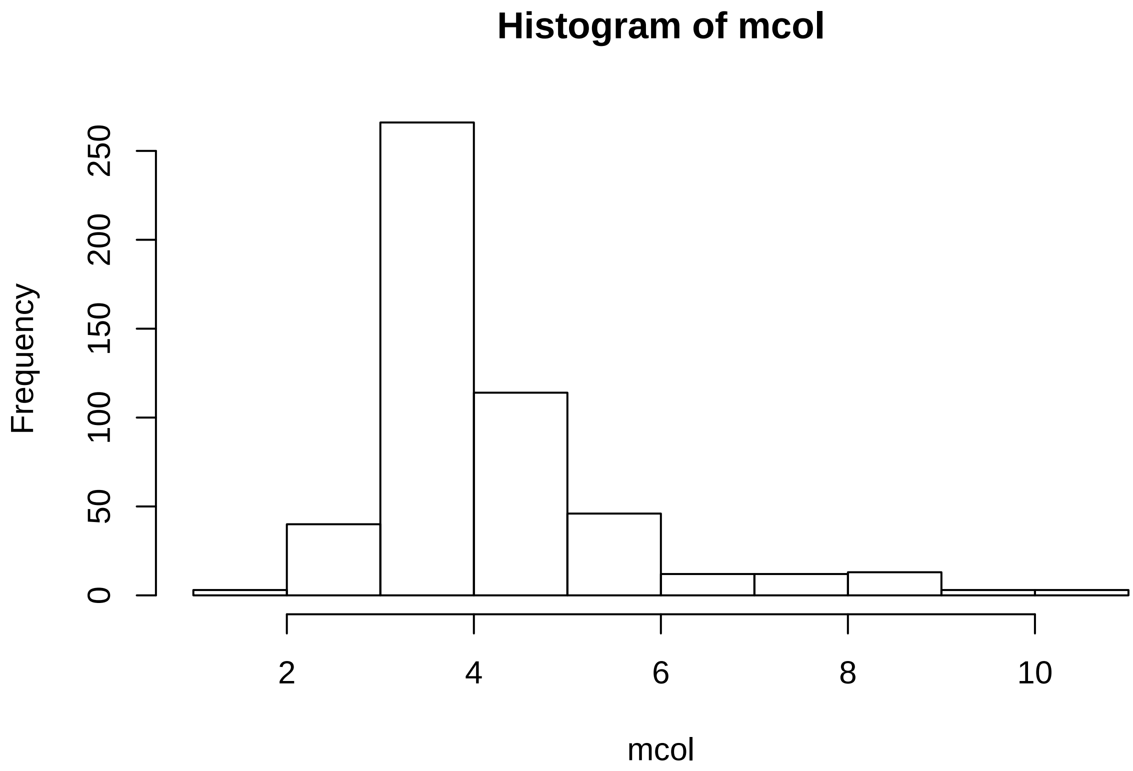
```
#Multicollinearity analysis
t1 = Sys.time()
library(car)
log.fit = glm(Y.train ~ . , data=data.train, family="binomial")
mcol = vif(log.fit)
print(mcol[mcol>10])
```

```
##      V114     V425     V549
## 10.04276 10.39798 10.04690
```

```
# V114      V425      V549
#10.04276 10.39798 10.04690
which.max(mcol)
```

```
## V425
##  343
```

```
hist(mcol)
```



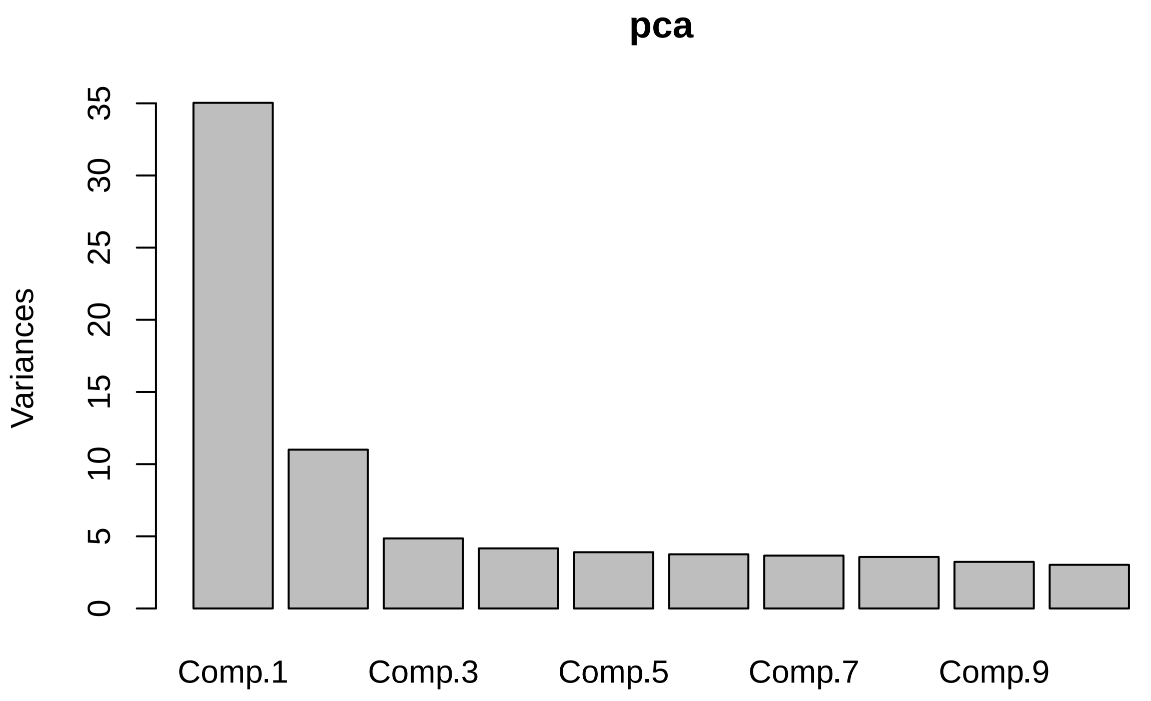
```
cat("Predictors with high variance inflation factor are ",which(mcol>10),"\n")
```

```
## Predictors with high variance inflation factor are  109 343 405
```

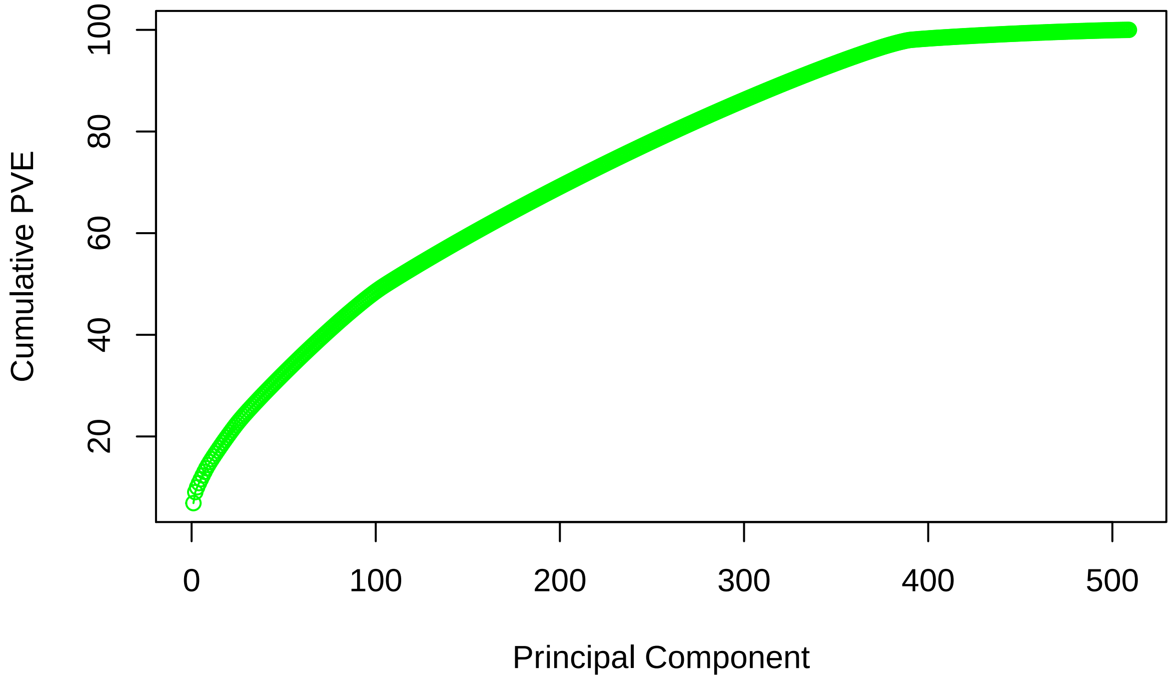
```
data.train = data.train[,-which.max(mcol)]
data.test = data.test[,-which.max(mcol)]
X = X[,-which(mcol>10)]
t2 = Sys.time()
print(t2-t1)
```

```
## Time difference of 38.11232 secs
```

```
#Dimensionality Reduction using PCA
t1=Sys.time()
pca = princomp(~ . , X)
plot(pca)
```



```
pve = 100*pca$sdev**2/sum(pca$sdev**2)
ndims = sum(cumsum(pve)<=90)
par(mfrow = c(1,1))
plot(cumsum(pve),type="o",ylab="Cumulative PVE",xlab="Principal Component",col="green")
```



```
#Since there is no clear elbow, cutoff of 90% is chosen, which gives 325 PCs
X.red = data.frame(pca$scores[,1:ndims])
X.train = X.red[train,]
X.test = X.red[-train,]
data.train = data.frame(X.train,Y.train)
data.test = data.frame(X.test, Y.test)
t2 = Sys.time()
print(t2-t1)
```

```
## Time difference of 11.73371 secs
```

```
#Feature Selection using LASSO
t1 = Sys.time()
library(glmnet)
```

```
## Loading required package: Matrix

## Loading required package: foreach

## Loaded glmnet 2.0-5
```

```
library(doMC)
```

```
## Loading required package: iterators
## Loading required package: parallel
```

```
library(pROC)
```

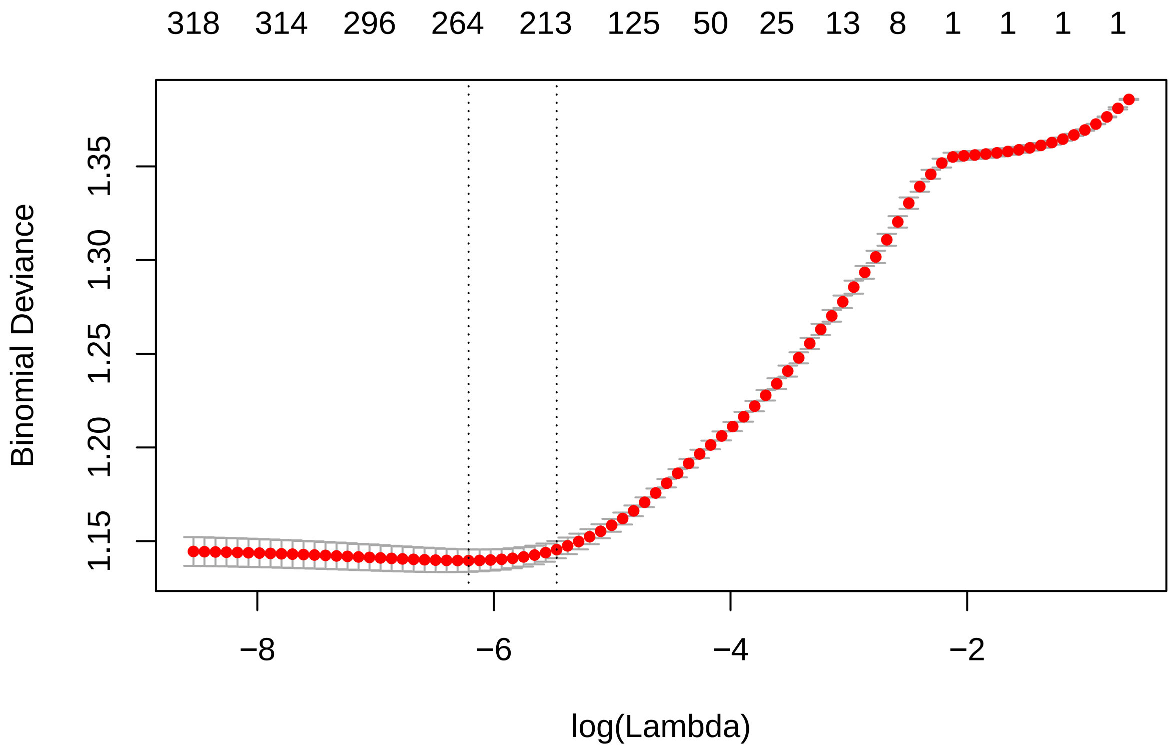
```
## Type 'citation("pROC")' for a citation.
```

```
##
## Attaching package: 'pROC'
```

```
## The following object is masked from 'package:glmnet':
##
##     auc
```

```
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
registerDoMC(cores=detectCores(T,T)-2)
set.seed(100)
Lasso.fits = cv.glmnet(as.matrix(X.train),Y.train,nfolds=5,alpha=1,standardize=F,intercept=FALSE,family=
plot(Lasso.fits)
```



```r
Lasso.lambda = Lasso.fits$lambda.min
cat("Chosen best lambda is ",Lasso.lambda,"\n")
```

```
## Chosen best lambda is  0.002000385
```

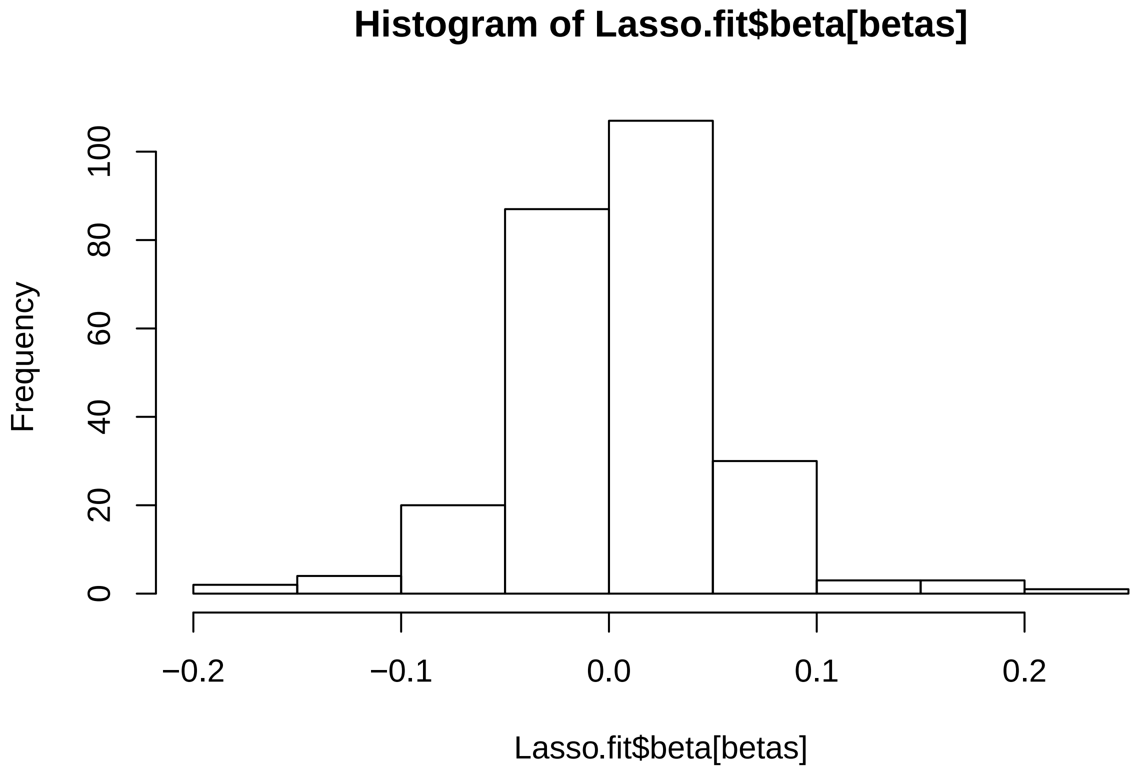
```r
Lasso.fit = glmnet(as.matrix(X.train),Y.train, alpha=1,standardize = F,lambda=Lasso.lambda,intercept = 0
cat("Number of features discarded=",sum(Lasso.fit$beta==0),"\n")
```

```
## Number of features discarded= 68
```

```r
betas = which(Lasso.fit$beta != 0)
cat("Number of predictors now are",length(betas),"\n")
```

```
## Number of predictors now are 257
```

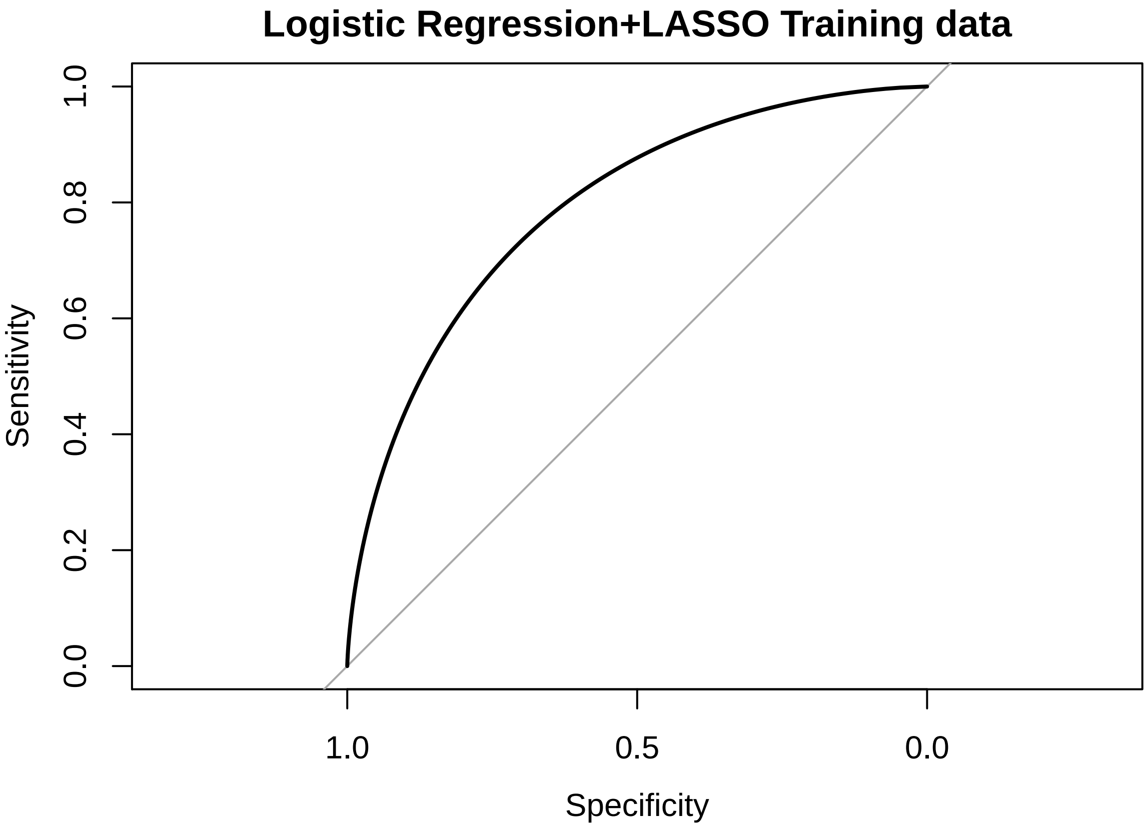
```r
# X.train = X.train[,betas]
# X.test = X.test[,betas]
# data.train = data.frame(X.train,Y.train)
# data.test = data.frame(X.test,Y.test)
hist(Lasso.fit$beta[betas])
```



```
#Training Error Rate
lasso.train = predict(Lasso.fit,type="response",newx=as.matrix(X.train))
lasso.train = as.numeric(lasso.train)
rocobj = roc(response=Y.train, predictor=lasso.train,smooth=T,auc=T)
cat("Area Under the Curve for logisitic regression+LASSO training data is ",rocobj$auc)
```

```
## Area Under the Curve for logisitic regression+LASSO training data is  0.791014
```

```
plot(rocobj,main="Logistic Regression+LASSO Training data")
```



```
lasso.train[lasso.train>=0.5] = 1
lasso.train[lasso.train<0.5] = 0
table(lasso.train, Y.train)
```

```
##            Y.train
## lasso.train    0    1
##           0 5434 2146
##           1 2050 5370
```

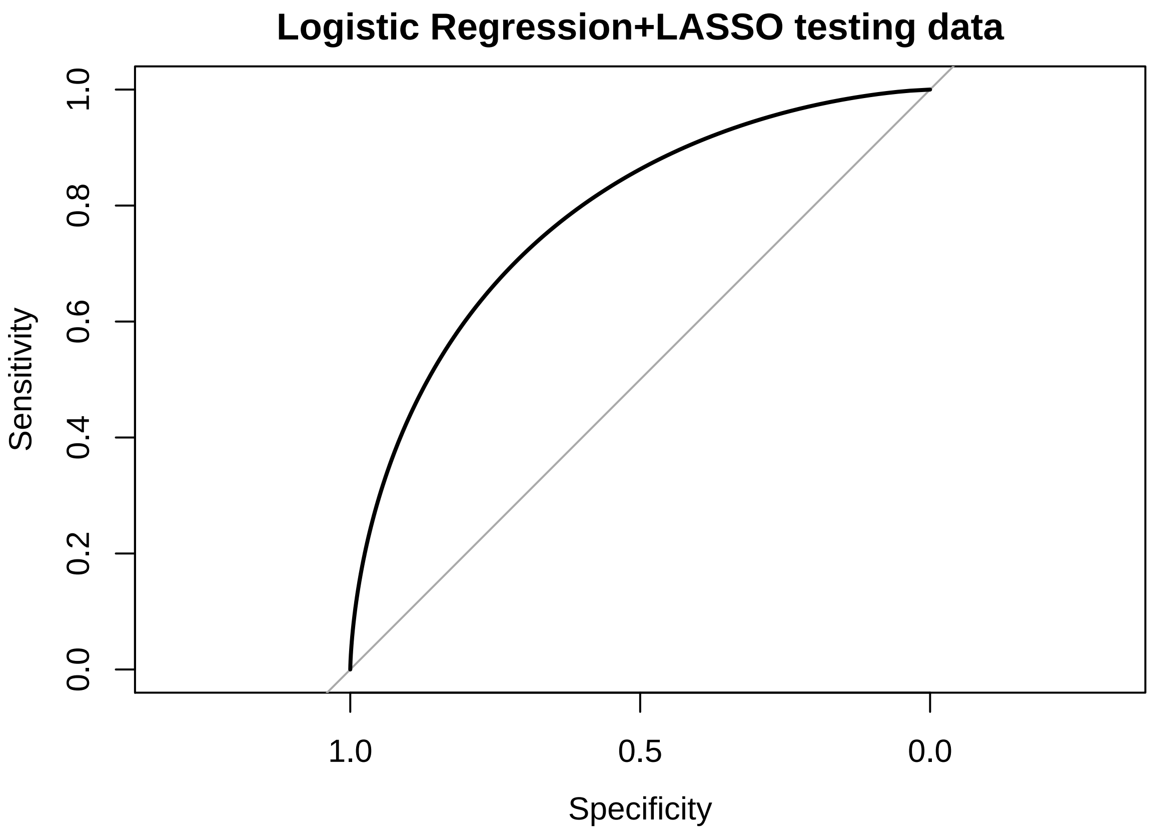
```
cat("Logistic Regression+LASSO Training Error Rate =",mean(lasso.train!=Y.train))
```

```
## Logistic Regression+LASSO Training Error Rate = 0.2797333
```

```
#Test Error Rate
lasso.test = predict(Lasso.fit, type="response",newx=as.matrix(X.test))
lasso.test = as.numeric(lasso.test)
rocobj = roc(response=Y.test, predictor=lasso.test,smooth=T,auc=T)
cat("Area Under the Curve for logisitic regression+LASSO testing data is ",rocobj$auc)
```

```
## Area Under the Curve for logisitic regression+LASSO testing data is  0.7815972
```

```
plot(rocobj,main="Logistic Regression+LASSO testing data")
```



```
lasso.test[lasso.test>=0.5] = 1
lasso.test[lasso.test<0.5] = 0
table(lasso.test, Y.test)
```

```
##           Y.test
## lasso.test    0    1
##          0 1834  740
##          1  682 1744
```

```
cat("Logistic Regression+LASSO testing Error Rate =",mean(lasso.test!=Y.test))
```

```
## Logistic Regression+LASSO testing Error Rate = 0.2844
```

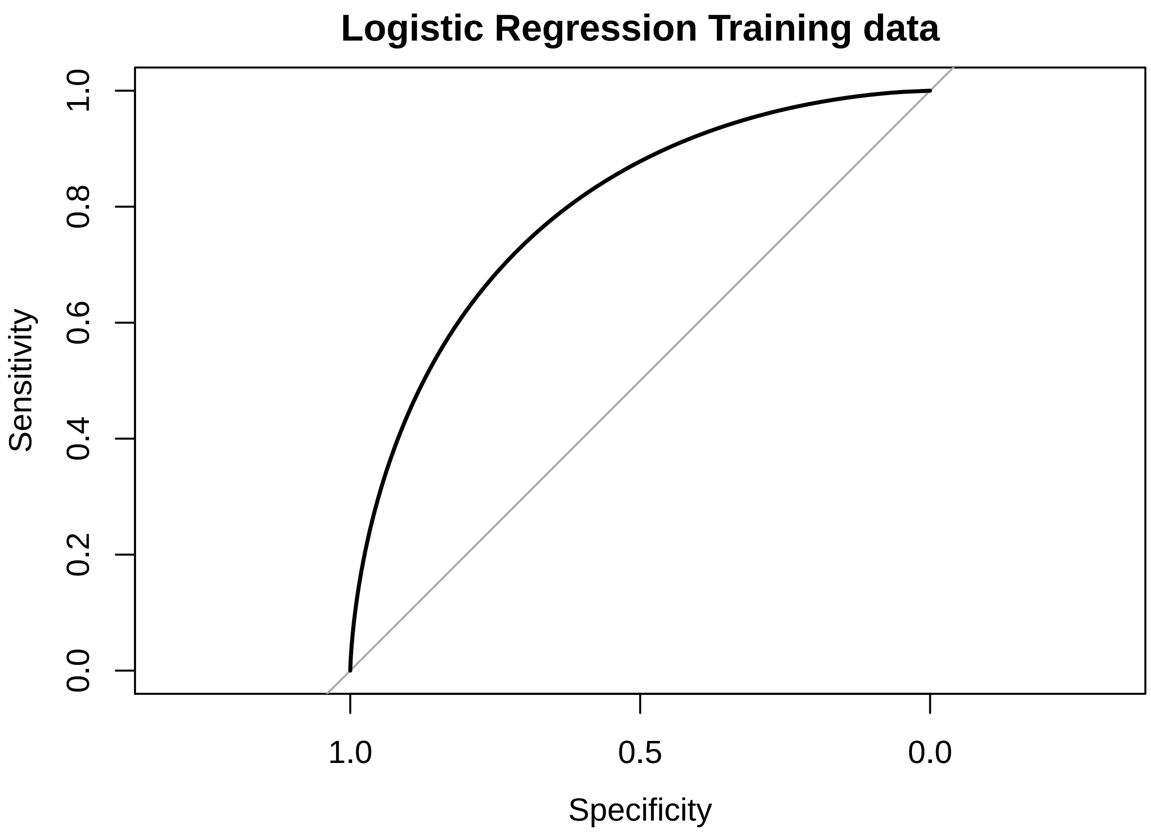
```r
t2 = Sys.time()
print(t2-t1)
```

```
## Time difference of 9.759658 secs
```

```r
#Logistic Regression
t1 = Sys.time()
set.seed(100)
log.fit = glm(Y.train ~ . ,data=data.train, family="binomial")
#Training Error
log.train = predict(log.fit,type="response")
rocobj = roc(response=Y.train, predictor=log.train,smooth=T,auc=T)
cat("Area Under the Curve for logisitic regression training data is ",rocobj$auc)
```

```
## Area Under the Curve for logisitic regression training data is  0.7926735
```

```r
plot(rocobj,main="Logistic Regression Training data")
```



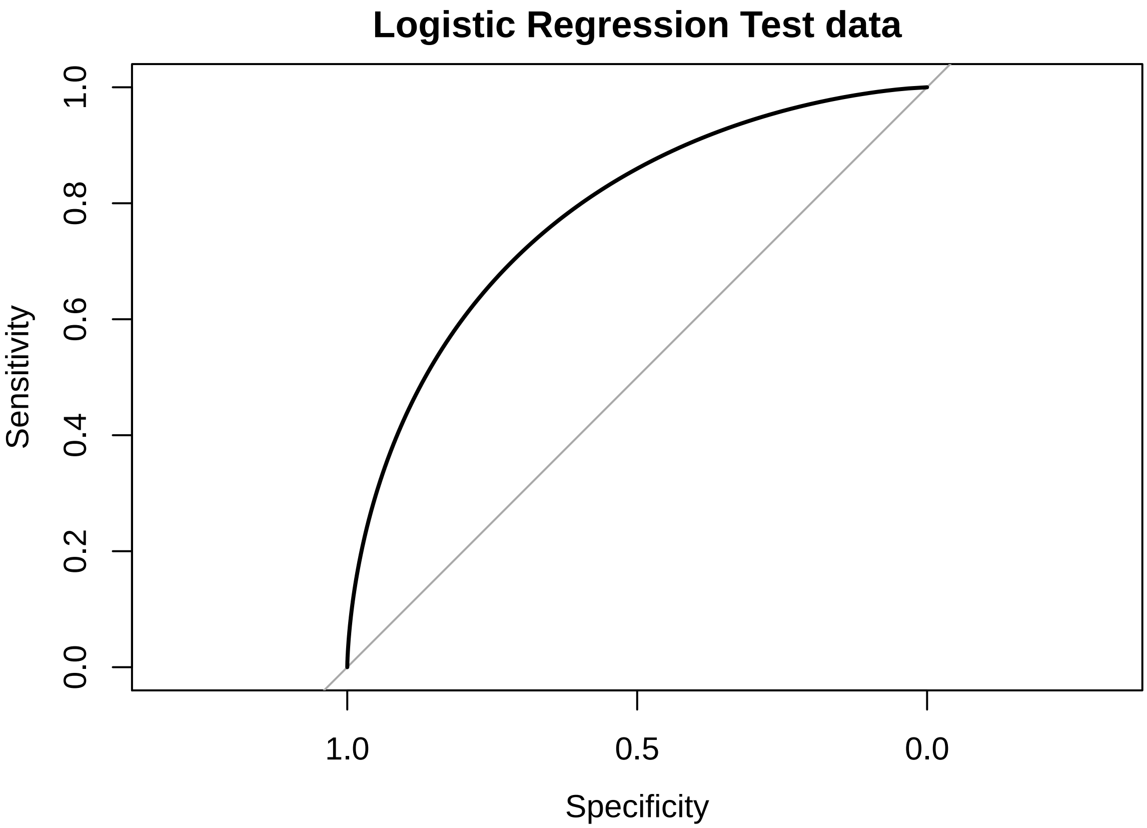
```r
log.train[log.train>=0.5] = 1
log.train[log.train<0.5] = 0
table(log.train, Y.train)
```

```
##          Y.train
## log.train    0    1
##         0 5409 2097
##         1 2075 5419
```

```
cat("Logistic Regression Training Error Rate =",mean(log.train!=Y.train))
```

```
## Logistic Regression Training Error Rate = 0.2781333
```

```
#Test Error
log.test = predict(log.fit,newdata = data.test,type="response")
rocobj=roc(response=Y.test,predictor=log.test,smooth=T,auc=T)
plot(rocobj,main="Logistic Regression Test data")
```



```
cat("Area Under the Curve for logisitic regression test data is ",rocobj$auc)
```

```
## Area Under the Curve for logisitic regression test data is  0.7800547
```

```
log.test[log.test<0.5] = 0
log.test[log.test!=0]=1
table(log.test,Y.test)
```

```
##         Y.test
## log.test    0    1
##        0 1818  726
##        1  698 1758
```

```r
cat("Logistic Regression Test Error Rate = ",mean(log.test!=Y.test),"\n")
```

```
## Logistic Regression Test Error Rate =  0.2848
```

```r
t2 = Sys.time()
print(t2-t1)
```

```
## Time difference of 8.994679 secs
```

```r
#Discriminant Analysis
t1 = Sys.time()
library(klaR)
```

```
## Loading required package: MASS
```

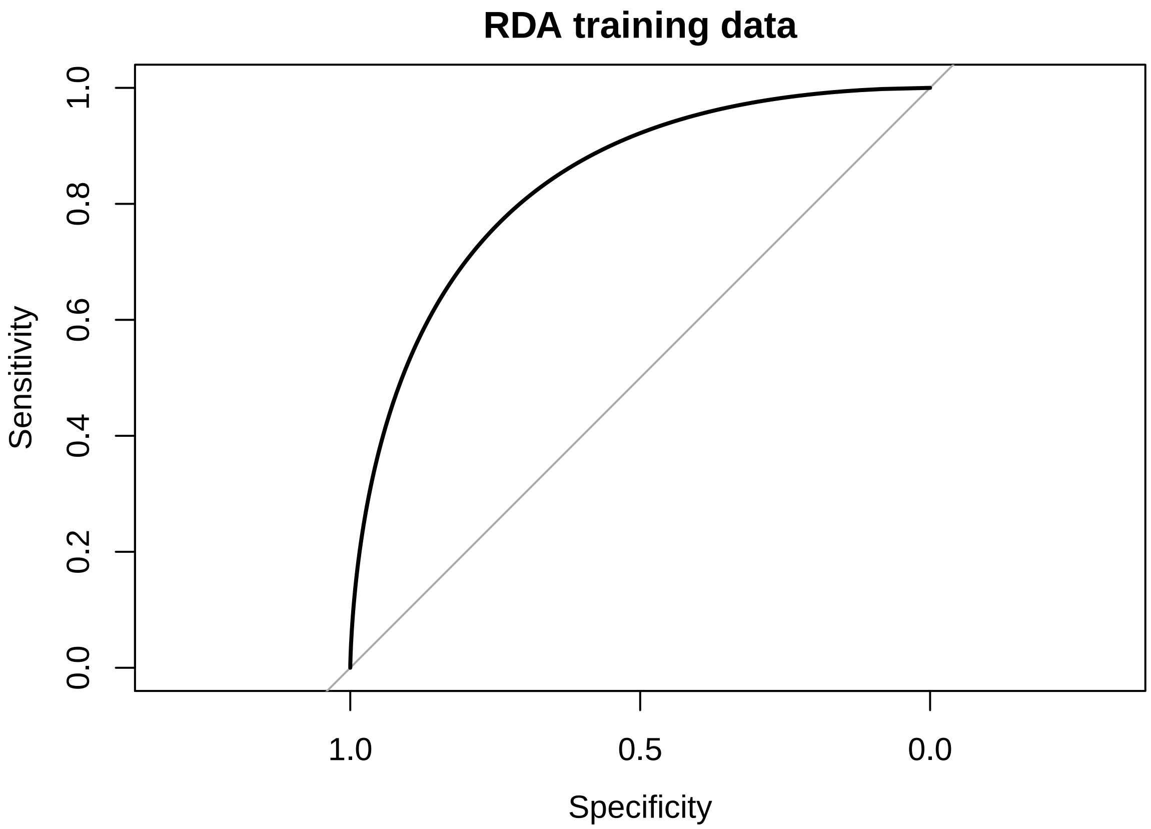
```r
set.seed(100)
rda.fit = rda(formula=formula(log.fit),data=data.train)
cat("Regularization parameters are ",rda.fit$regularization,"\n")
```

```
## Regularization parameters are  0.02738831 0.9807333
```

```r
cat("RDA cross-validated Training Error Rate = ",rda.fit$error.rate[2],"\n")
```

```
## RDA cross-validated Training Error Rate =  0.2904047
```

```r
#Training Error Rate
rda.train = predict(rda.fit, type="response")
rda.train = as.numeric(rda.train$posterior[,"1"])
rocobj = roc(response=Y.train, predictor=rda.train,smooth=T,auc=T)
plot(rocobj,main="RDA training data")
```



```
cat("AUC for RDA training data is",rocobj$auc,"\n")
```

```
## AUC for RDA training data is 0.8353954
```

```
rda.train[rda.train<0.5]=0
rda.train[rda.train!=0]=1
table(rda.train,Y.train)
```

```
##          Y.train
## rda.train    0    1
##         0 5706 1856
##         1 1778 5660
```

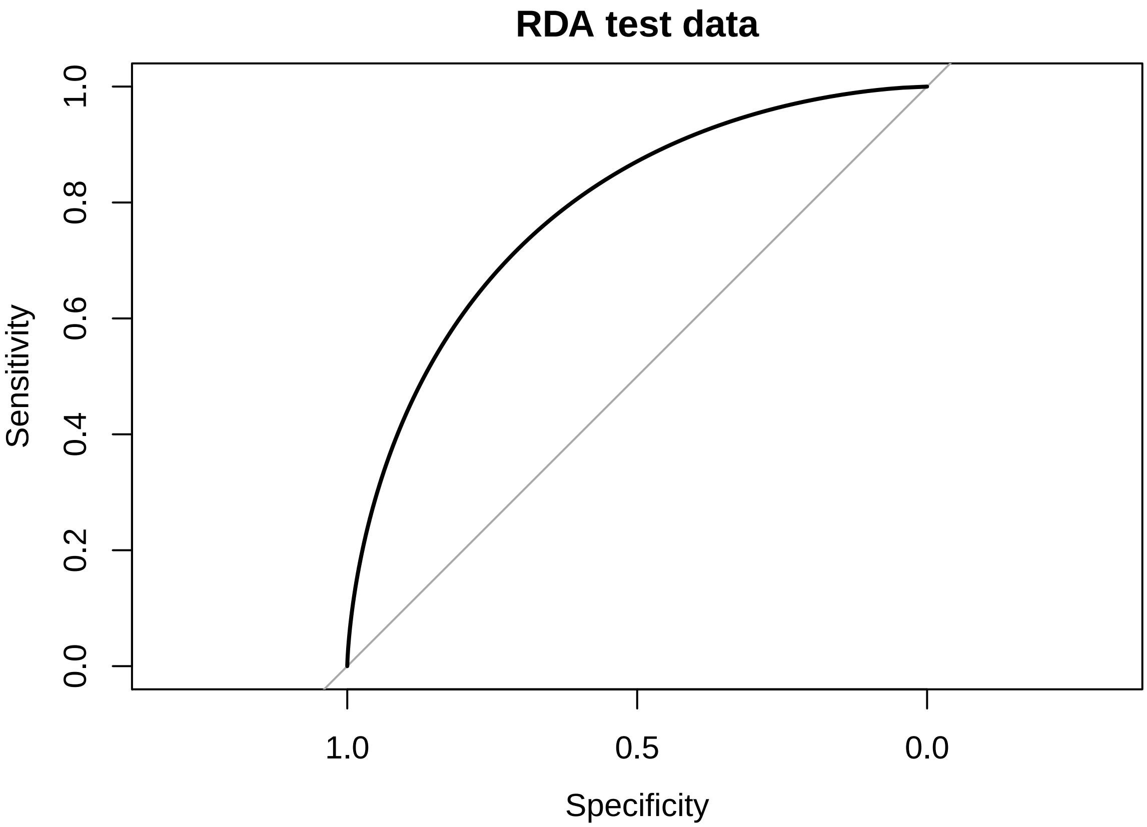
```
cat("RDA training Error Rate is ",mean(rda.train!=Y.train),"\n")
```

```
## RDA training Error Rate is  0.2422667
```

```
#Test Error Rate
rda.test = predict(rda.fit,newdata=data.test,type="response")
rda.test = as.numeric(rda.test$posterior[,"1"])
rocobj=roc(response=Y.test,predictor=rda.test,smooth=T,auc=T)
cat("AUC for RDA test data is ",rocobj$auc,"\n")
```

```
## AUC for RDA test data is  0.7861497
```

```
plot(rocobj,main="RDA test data")
```



```
rda.test[rda.test<0.5]=0
rda.test[rda.test!=0]=1
table(rda.test,Y.test)
```

```
##          Y.test
## rda.test    0    1
##        0 1825  731
##        1  691 1753
```

```
cat("RDA test error rate = ",mean(rda.test!=Y.test),"\n")
```

```
## RDA test error rate =  0.2844
```

```
t2 = Sys.time()
print(t2-t1)
```

```
## Time difference of 20.29742 mins
```