MRR_TP3_Zeyu_CHEN_Clement_VEYSSIERE

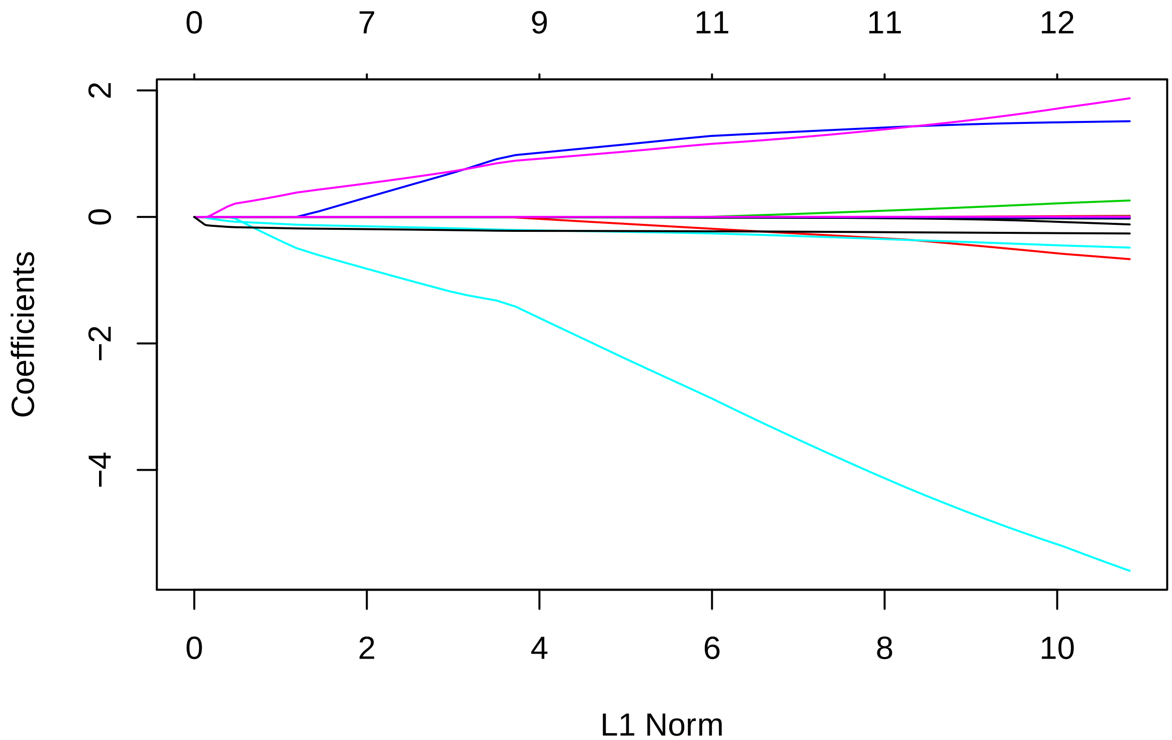
Zeyu CHEN

2018/11/11

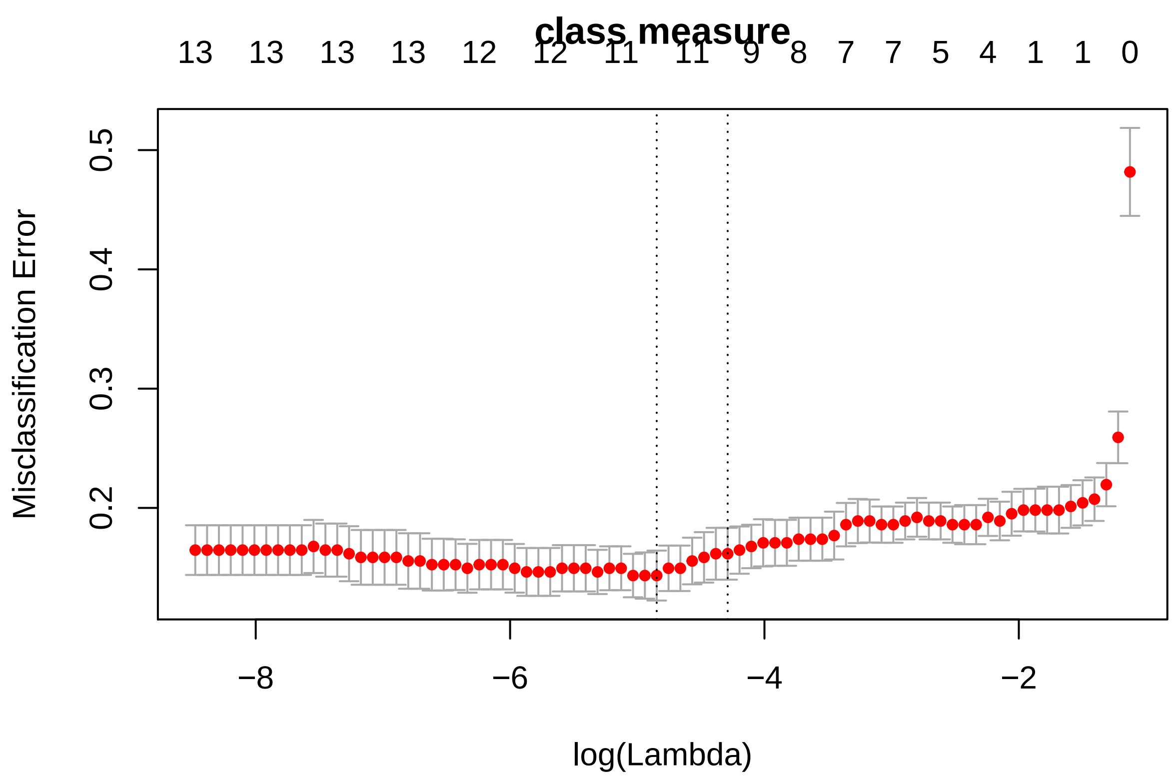
```
library(MASS)
options(warn=-1)
library(glmnet)
```

```
## Loading required package: Matrix

## Loading required package: foreach

## Loaded glmnet 2.0-16
```

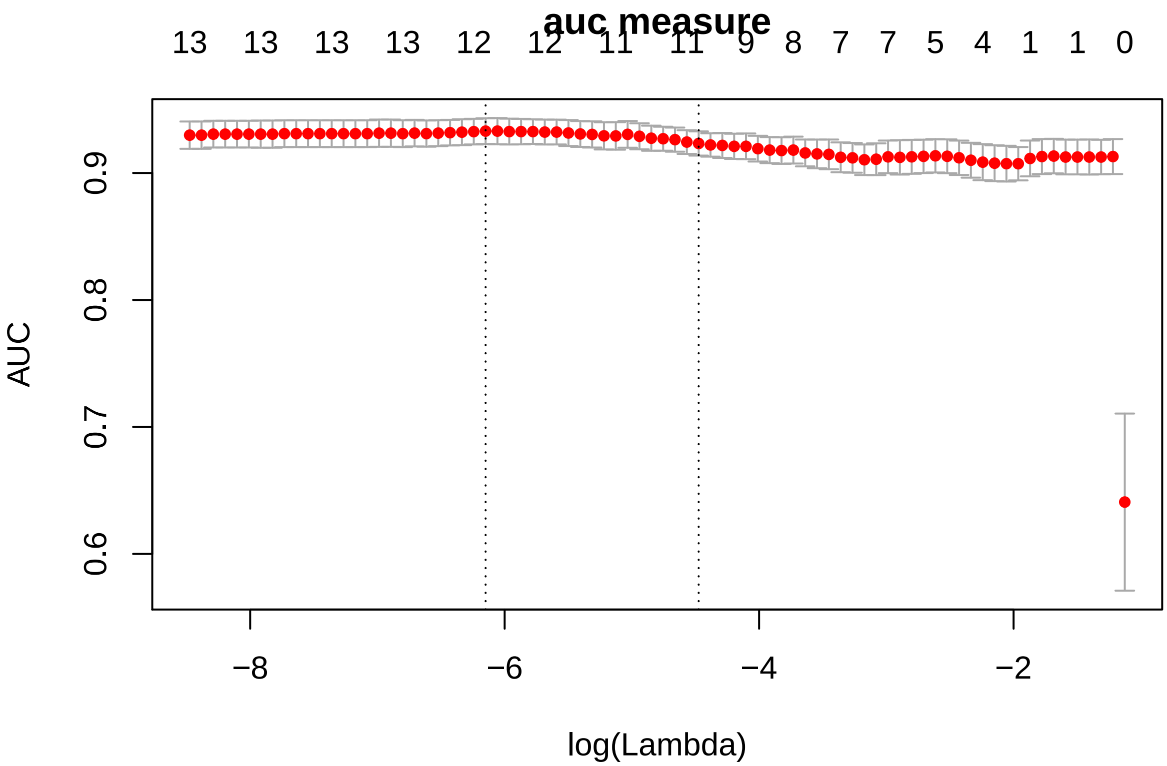
```
library(mlbench)
#load the data
data(BostonHousing)
medvBin <- as.numeric(BostonHousing$medv>median(BostonHousing$medv))
BostonHousing$medv <- medvBin

set.seed(100)

# partitionning
sub <- sample(nrow(BostonHousing), 0.65 * nrow(BostonHousing))
tabTrain <- BostonHousing[sub,]
tabTest <- BostonHousing[-sub,]

#drop the medv and transforme to matrix
X <- data.matrix(subset(tabTrain,select= -medv))


#Using lasso regression with alpha = 1
glmmod <- glmnet(X,as.factor(tabTrain$medv),alpha = 1,family="binomial")
plot(glmmod)
```



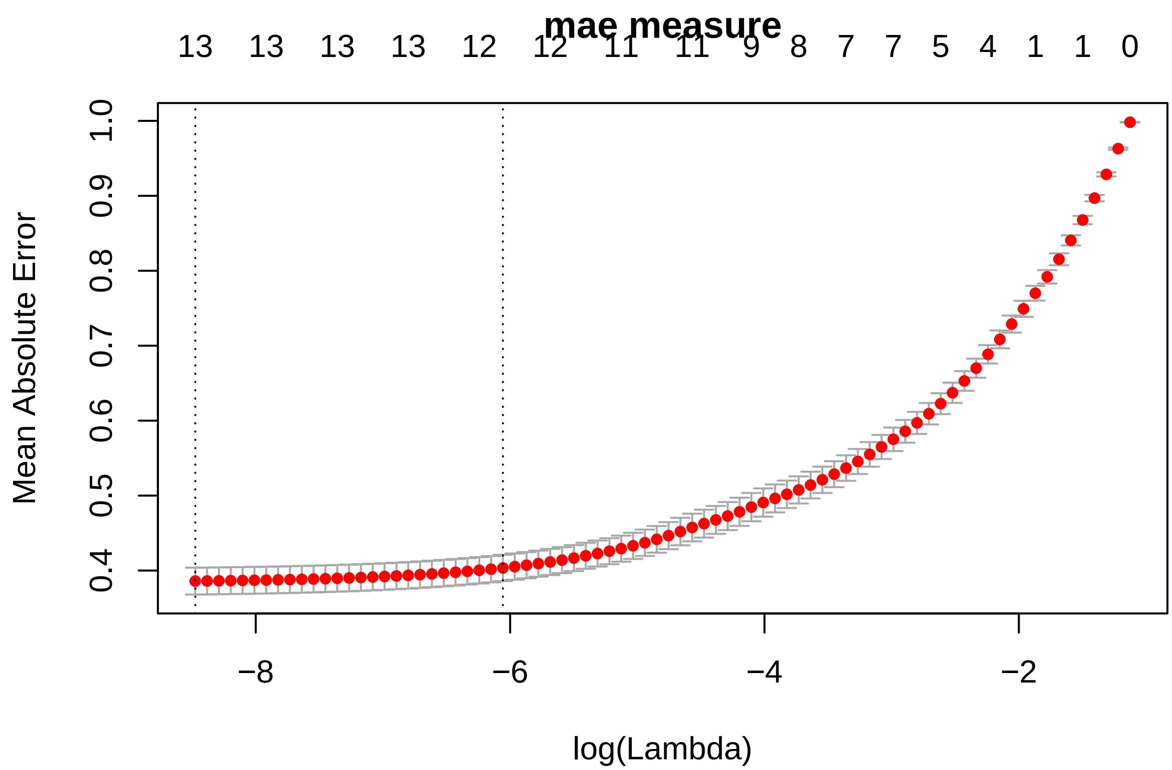

```r
#We can find the samller L1 norm is , the more coef are equal to 0

#Using lasso regression through 10-fold with type.measure="class"
modLassoC<- cv.glmnet(X,tabTrain$medv,family="binomial",type.measure="class",alpha=1)
plot(modLassoC,main="class measure")
```



```r
#Using lasso regression through 10-fold with type.measure="auc"
modLassoA<- cv.glmnet(X,tabTrain$medv,family="binomial",type.measure="auc",alpha=1)
plot(modLassoA,main="auc measure")
```




```r
#Using lasso regression through 10-fold with type.measure="mae"
modLassoM<- cv.glmnet(X,tabTrain$medv,family="binomial",type.measure="mae",alpha=1)
plot(modLassoM,main="mae measure")
```




```
lambda.min<-vector(length = 3)
lambda.1se<-vector(length = 3)
newx <- data.matrix(subset(tabTest,select=-medv))
#Predicting with ="modLassoC" and with "s=modLassoC$lambda.min"
preMin<- predict(modLassoC,newx =newx,s=modLassoC$lambda.min,type = "response")
lambda.min[1] <- sum((as.numeric(preMin>0.5)-tabTest$medv)^2)

#Predicting with ="modLassoC" and with "s=modLassoC$lambda.1se"
pre1se<- predict(modLassoC,newx =newx,s=modLassoC$lambda.1se,type = "response")
lambda.1se[1] <- sum((as.numeric(pre1se>0.5)-tabTest$medv)^2)

#Predicting with ="modLassoA" and with "s=modLassoA$lambda.min"
preMin<- predict(modLassoA,newx =newx,s=modLassoA$lambda.min,type = "response")
lambda.min[2] <- sum((as.numeric(preMin>0.5)-tabTest$medv)^2)

#Predicting with ="modLassoA" and with "s=modLassoA$lambda.1se"
pre1se<- predict(modLassoA,newx =newx,s=modLassoA$lambda.1se,type = "response")
lambda.1se[2] <- sum((as.numeric(pre1se>0.5)-tabTest$medv)^2)

#Predicting with ="modLassoM" and with "s=modLassoM$lambda.min"
preMin<- predict(modLassoM,newx =newx,s=modLassoM$lambda.min,type = "response")
lambda.min[3] <- sum((as.numeric(preMin>0.5)-tabTest$medv)^2)

#Predicting with ="modLassoM" and with "s=modLassoM$lambda.1se"
pre1se<- predict(modLassoM,newx =newx,s=modLassoM$lambda.1se,type = "response")
lambda.1se[3] <- sum((as.numeric(pre1se>0.5)-tabTest$medv)^2)


data.frame(lambda.min,lambda.1se,row.names = c("Class wrong times ","Auc wrong times ",
                                               "Mae wrong times "))
```

```
##                    lambda.min lambda.1se
## Class wrong times         22         24
## Auc wrong times           20         22
## Mae wrong times           19         20
```

```
# We can find that the performence of Mae methods is quiet good. And the performence using
# lambda.min is better than using lambda.1se.
```