HW 8

Anish Mohan

December 2, 2015

For the ALS project, I did the following steps:

1. Get the training features and replace the missing entries with median values.
2. Use Lasso with Cross validation to get the the lambda that gives the minimum RMSE.
3. Get the data for the leaderboard and replace the missing entries with median values.
4. Predict the ALFRS_Slope value, using the parameters from lasso and the best lambda.

Code:

```
#Getting the training features and target

#Get the input target or Y i.e the response variable
train_targ<-read.csv("training_target.csv")
print(paste0("Number of patients: ", nrow(train_targ)))
```

```
## [1] "Number of patients: 2424"
```

```
#get the input features or X
train_feat<-read.csv("training_features.csv")
print(paste0("Number of features: ", ncol(train_feat)))
```

```
## [1] "Number of features: 858"
```

```
#summary of training target
summary(train_targ["ALSFRS_slope"])
```

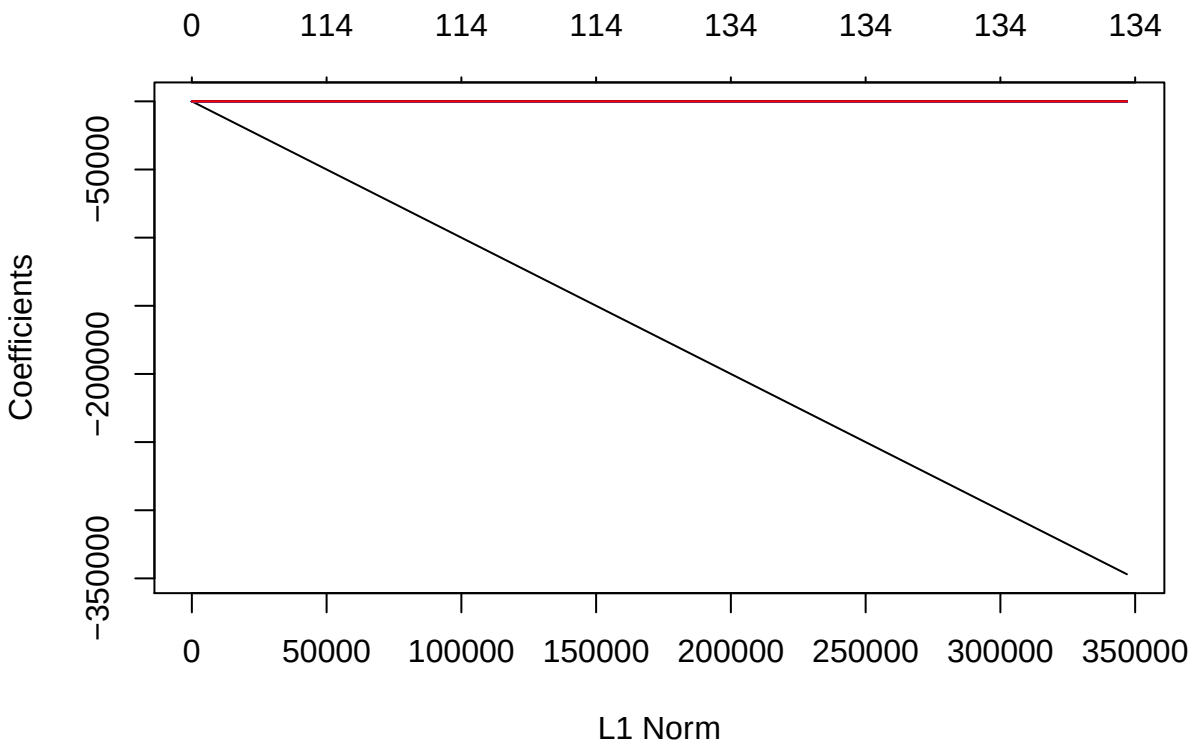
```
##   ALSFRS_slope
##  Min.   :-4.3452
##  1st Qu.:-1.0863
##  Median :-0.6207
##  Mean   :-0.7308
##  3rd Qu.:-0.2742
##  Max.   : 1.2070
```

```
#function to count the number of empty entries in an inputx
numnas<-function(x){
  sum_na<-0
  for (n in x){
    if(is.na(n)==T)
      sum_na=sum_na+1
```

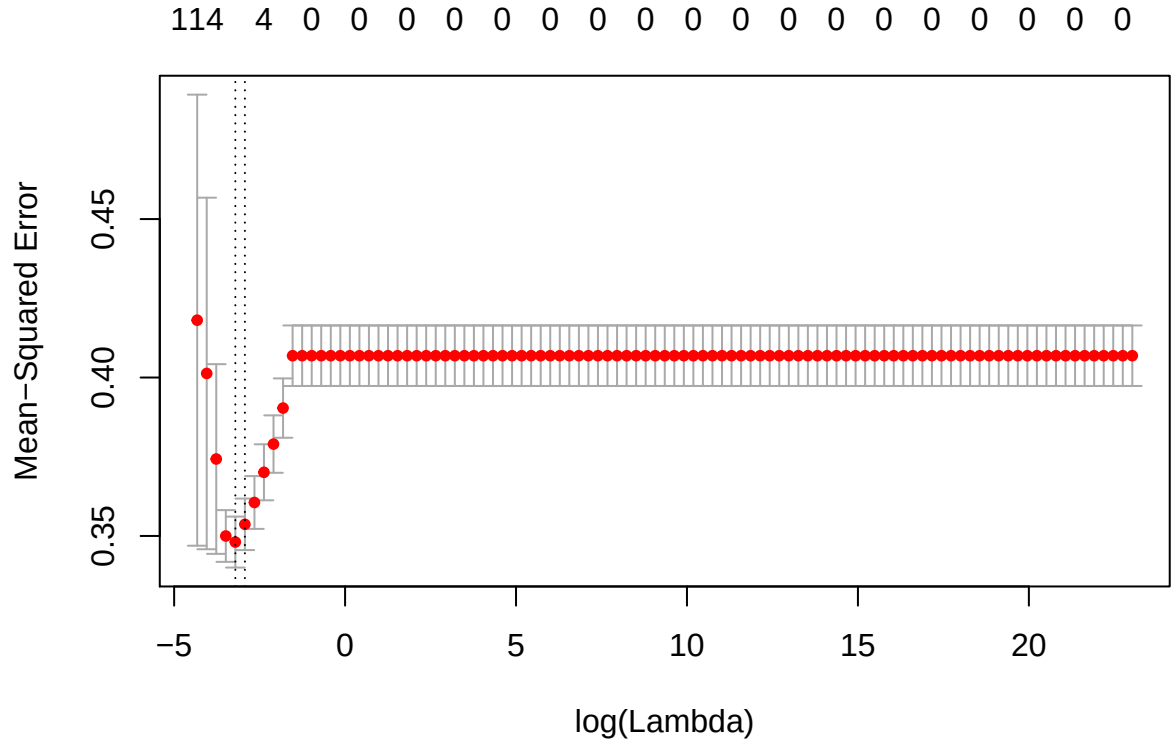
```
  }
  return(sum_na)
}

#Running the function on training features gives us the number of entries in a column that are empty.
num_nas_col=apply(train_feat,2,numnas)

#Function to get columns with a greater number than 'a' empty entries
for( i in names(num_nas_col)){
  if(num_nas_col[i]<500){
    #print(i)
  }
}

#Missing data points are problem, creat an alternative data set with all missing values
# filled with median

feature.names<-names(train_feat)
temp=train_feat

for(feature.name in feature.names[-1]){
  dummy_name<-paste0("is.na.",feature.name)
  is.na.feature <-is.na(temp[,feature.name])
  temp[,dummy_name]<-as.integer(is.na.feature)
  temp[is.na.feature,feature.name]<-median(temp[,feature.name], na.rm=TRUE)
}

train_feat_median=temp[1:2424,1:858]
df.median=data.frame(ALFRS_slope=train_targ$ALSFRS_slope, train_feat_median)


set.seed(1)
train=sample(1:nrow(df.median),2000)
test=-train

df.median.test=df.median[test,]
df.median.train=df.median[train,]


library(glmnet)
```

```
## Loading required package: Matrix
## Loading required package: foreach
## Loaded glmnet 2.0-2
```

```
x=model.matrix(df.median.train$ALFRS_slope~.,df.median.train)[,-1]
y=df.median.train$ALFRS_slope

grid=10^seq(10,-2,length=100)
lasso.mod=glmnet(x,y, alpha=1, lambda=grid)

plot(lasso.mod)
```




```
#using cross validation
set.seed(1)
cv.out=cv.glmnet(x,y,alpha=1,lambda=grid)
plot(cv.out)
```



```r
bestlam=cv.out$lambda.min


df.test.x=model.matrix(df.median.test$ALFRS_slope~.,df.median.test)[,-1]
df.test.y=df.median.test$ALFRS_slope

lasso.pred=predict(lasso.mod,s=bestlam,newx=df.test.x)
mean((lasso.pred-df.test.y)^2)
```

```
## [1] 0.273334
```

```r
#Preparing for leaderboard predictions:
lb_feat<-read.csv("leaderboard_features.csv")
leaderboard.predictions <- read.csv("leaderboard_predictions-example.csv")

num_nas_col=apply(lb_feat,2,numnas)

feature.names<-names(lb_feat)
temp=lb_feat

for(feature.name in feature.names[-1]){
  dummy_name<-paste0("is.na.",feature.name)
  is.na.feature <-is.na(temp[,feature.name])
  temp[,dummy_name]<-as.integer(is.na.feature)
  ifelse (is.na(median(temp[,feature.name]))==F, temp[is.na.feature,feature.name]<-median(temp[,feature


}

lb_feat_median=temp[1:187,1:858]

lb.median=data.frame(ALFRS_slope=leaderboard.predictions$ALSFRS_slope, lb_feat_median)

lb.x=model.matrix(lb.median$ALFRS_slope~.,lb.median)[,-1]
lb.y=lb.median$ALFRS_slope

lasso.pred=predict(lasso.mod,s=bestlam,newx=lb.x)
leaderboard.predictions$ALSFRS_slope=lasso.pred
write.csv(leaderboard.predictions, file = "leaderboard_predictions.csv",row.names=FALSE)
```