Code

```
library(data.table)
library(plyr)

dona <- read.csv('donation data.csv',sep = ',', header = T)
code <- read.csv('dmef1code.csv',sep = ',', header = T)
regions <- read.csv('usregions.csv',sep = ',', header = T)

IFTEST <- (1:length(dona$TARGDOL))%%3==0
IFDONATE <- dona$TARGDOL>0
dona <- cbind(dona,IFTEST,IFDONATE)

dona <- merge(x = dona, y = regions, by.x = 'STATCODE', by.y = 'State.Code', all.x = TRUE)

dona$Region[is.na(dona$Region)] <- "Sparse"
dona$Sub.Region[is.na(dona$Sub.Region)] <- "Sparse"

dona1 <- merge(x = dona, y = code, by.x = 'CNCOD1', by.y = 'CODE', all.x = TRUE)
dona1 <- merge(x = dona1, y = code, by.x = 'CNCOD2', by.y = 'CODE', all.x = TRUE)
dona1 <- merge(x = dona1, y = code, by.x = 'CNCOD3', by.y = 'CODE', all.x = TRUE)
dona1 <- rename(dona1, c('CODETYPE.x' = 'CNTYPE1','CODETYPE.y' = 'CNTYPE2','CODETYPE'= 'CNTYPE3'))

dona1 <- merge(x = dona1, y = code, by.x = 'SLCOD1', by.y = 'CODE', all.x = TRUE)
dona1 <- merge(x = dona1, y = code, by.x = 'SLCOD2', by.y = 'CODE', all.x = TRUE)
dona1 <- merge(x = dona1, y = code, by.x = 'SLCOD3', by.y = 'CODE', all.x = TRUE)
dona1 <- rename(dona1, c('CODETYPE.x' = 'SLTYPE1','CODETYPE.y' = 'SLTYPE2','CODETYPE'= 'SLTYPE3'))

dona1[,37]<-dona1$CNTRLIF/dona1$CNTMLIF
names(dona1)[37]<-"averageamount"

dona1[,38]<-dona1$CNMONF/dona1$CNTMLIF
names(dona1)[38]<-"frequency"

dona2<-subset(dona1,select=-c(SLCOD3,CNCOD3,CNCOD2,CNDAT2,CNDAT3,CNCOD1,SLCOD1,SLCOD2,CNDOL2,CNDOL3,CNM
write.csv(dona1,"12012120.csv")

donate.data = read.csv('~/401-Project1/Data/12012120.csv')
donate.data = cbind(donate.data,rep(0,length(donate.data$X)))
names(donate.data)[40]='log_target'
donate.data$log_target[donate.data$TARGDOL>0]=log10(donate.data$TARGDOL[donate.data$TARGDOL>0])

# frequency recalculate:
donate.data$frequency = (donate.data$CNMON1+1)/donate.data$CNTMLIF

# LOG: CONLARGE CNTRLIF CONTRFST CNMONF CNMONL
donate.data = cbind(donate.data,log(donate.data$CONLARG),log(donate.data$CNTRLIF),log(donate.data$CONTRI

names(donate.data)[c(41,42,43,44,45)]=c('log_conlarg','log_cntrlif','log_contrfst','log_cnmonf','log_cn
donate.data$log_contrfst=log(donate.data$CONTRFST+1)
# CNTMLIF  1  2  3 4-10 >10
donate.data = cbind(donate.data,rep(0,length(donate.data$X)))
```

```
names(donate.data)[46]='fac_cntmlif'



for (i in 1:length(donate.data$X)){


  if (donate.data$CNTMLIF[i]==1){
    donate.data$fac_cntmlif[i]=1
  }
  else if (donate.data$CNTMLIF[i]==2){
    donate.data$fac_cntmlif[i]=2
  }
  else if (donate.data$CNTMLIF[i]==3){
    donate.data$fac_cntmlif[i]=3
  }
  else if (donate.data$CNTMLIF[i]>=4 && donate.data$CNTMLIF[i]<=10 ){
    donate.data$fac_cntmlif[i]=4
  }
  else {
    donate.data$fac_cntmlif[i]=5
  }
}

donate.data$fac_cntmlif = as.factor(donate.data$fac_cntmlif)


# log average/frequency
donate.data = cbind(donate.data,log(donate.data$averageamount),log(donate.data$frequency))
names(donate.data)[c(47,48)]=c('log_average','log_frequency')

donate.data=read.csv('~/401-Project1/Data/parseddata1201.csv')
donate.data$fac_cntmlif=as.factor(donate.data$fac_cntmlif)
donate.data=cbind(donate.data,donate.data$TARGDOL>=5)
names(donate.data)[50]='IF_TARGET_5'
donate.data=cbind(donate.data,donate.data$TARGDOL>5)
names(donate.data)[51]='IF_TARGETL5'

donate.data=cbind(donate.data,donate.data$TARGDOL>=10)
names(donate.data)[52]='IF_TARGET_10'

donate.data=cbind(donate.data,is.na(donate.data$CNDOL2),is.na(donate.data$CNDOL3))
names(donate.data)[53]='IF_MISSING_CNDOL2'
names(donate.data)[54]='IF_MISSING_CNDOL3'

donate.data$CNDOL2[is.na(donate.data$CNDOL2)]=0
donate.data$CNDOL3[is.na(donate.data$CNDOL3)]=0

donate.data$CNMON2[is.na(donate.data$CNMON2)]=170
donate.data$CNMON3[is.na(donate.data$CNMON3)]=170

#get the center and sd of each column
varlist = c('ID', "TARGDOL","IF_TARGET_5","IF_TARGETL5","IF_TARGET_10",'SLCOD1','CNDOL1','CNDOL2','CNDO
```

```
            "averageamount", "frequency",  "log_target",  "log_conlarg" , "log_cntrlif", "log_contrfst"
log_formula = 'X~1'
for (i in 1:length(varlist)){
  log_formula = paste(log_formula,varlist[i],sep="+")
}
log_formula=as.formula(log_formula)
matrixfirst=model.matrix(log_formula,data=donate.data)[,-1]
matrixfirst_train = matrixfirst[which(donate.data$IFTEST==0),]
matrixfirst_test = matrixfirst[donate.data$IFTEST==1,]

meanlist_for_standardize = colMeans(matrixfirst_train)
meanlist_for_standardize[1:5]=0
sdlist_for_standardize = sqrt(apply(matrixfirst_train,2,var))
sdlist_for_standardize[1:5]=1

#standardize both the train set and the test set
xxx=t(t(matrixfirst_train) - meanlist_for_standardize)
matrixfirst_train=t(t(xxx)/sdlist_for_standardize)
matrixfirst_train=as.data.frame(matrixfirst_train)
xxx=t(t(matrixfirst_test) - meanlist_for_standardize)
matrixfirst_test=t(t(xxx)/sdlist_for_standardize)
matrixfirst_test=as.data.frame(matrixfirst_test)

#write into csv
write.csv(matrixfirst_train,"~/401-Project1/Data/trainset_standardized.csv")
write.csv(matrixfirst_test,"~/401-Project1/Data/testset_standardized.csv")

# load in data
trainset = read.csv('~/401-Project1/Data/trainset_standardized.csv')

# resample the data to make the dataset balanced
ntrain_t = sum(trainset$TARGDOL>0)
resample=trainset[(trainset$TARGDOL>0)==F, ]
set.seed(401)
resample=rbind(resample[sample(1:length(resample$TARGDOL),ntrain_t),],trainset[trainset$TARGDOL>0,])



# LIST OF VARIABLES
varlist = names(trainset)

# drop responses and IDs
varlist = varlist[-c(1,2,3,4,5,6,23,24,40)]

log_formula = 'TARGDOL~1'
for (i in 1:length(varlist)){
  log_formula = paste(log_formula,varlist[i],sep="+")
}

#formula
log_formula=as.formula(log_formula)

# lm and hatvalues
```

```
fit = lm(log_formula,data=resample)
p=length(fit$coefficients)
n=length(resample$TARGDOL)
h=hatvalues(fit)
h=as.data.frame(h)
hratio = 4

threshold = hratio*(p+1)/n
excludelistID = resample$ID[h>threshold]

resample = resample[-excludelistID, ]

write.csv(resample,'~/401-Project1/Data/trainset_resampled_noinfluential_standardized.csv')
write.csv(excludelistID,'~/401-Project1/Data/influentialID_in_resample.csv')


# FOR MULTILINEAR MODEL , lm and hatvalues
lmtrainset = subset(trainset,trainset$TARGDOL>0)
# in case if you want to include some target = 0 sample
set.seed(401)
lmtrainset = rbind(trainset[sample(1:sum(trainset$TARGDOL==0),round(0.1*sum(trainset$TARGDOL>0))),  ],lm

lmfit = lm(log_formula,data=lmtrainset)

p=length(lmfit$coefficients)
n=length(lmtrainset$TARGDOL)
h=hatvalues(lmfit)
h=as.data.frame(h)
hratio = 4

threshold = hratio*(p+1)/n
excludelistID_lm = lmtrainset$ID[h>threshold]

lmtrainset = lmtrainset[-excludelistID_lm, ]

write.csv(lmtrainset,'~/401-Project1/Data/trainset_multilinear_noinfluentialnoimputed_standardized.csv')
write.csv(excludelistID_lm,'~/401-Project1/Data/influentialID_noimputed_multilinear.csv')

# load in data
trainset = read.csv('~/401-Project1/Data/trainset_resampled_noinfluential_standardized.csv')
testset = read.csv('~/401-Project1/Data/testset_standardized.csv')

trainset=trainset[,-1]

# LIST OF VARIABLES
varlist = names(trainset)

# drop responses and IDs
varlist = varlist[-c(1,2,3,4,5,6,23,24,40)]

log_formula = 'TARGDOL~1'
for (i in 1:length(varlist)){
  log_formula = paste(log_formula,varlist[i],sep="+")
```

```
}


#add quadratic
for (i in 1:length(varlist)){
  log_formula = paste(log_formula,'+I(',varlist[i],'^2)')
}

#add interaction terms
for (i in 1:(length(varlist)-1)){

  for (j in (i+1):length(varlist)){
    log_formula=paste(log_formula,paste(varlist[i],varlist[j],sep='*'),sep='+')
  }
}

#formula
log_formula=as.formula(log_formula)



#generate the model as input for glmnet function
trainmatrix= model.matrix(log_formula,data=trainset)

testmatrix = model.matrix(log_formula,data=testset)



# donate.regtrainmatrix = model.matrix(log_formula,data=donate.trainforreg)
library(glmnet)
cv.regression = cv.glmnet(x=trainmatrix,y=trainset$IF_TARGET_5TRUE,alpha=1,family = 'binomial' ,nfolds =
plot(cv.regression)
donate.cvregmodel = glmnet(x=trainmatrix,y=trainset$IF_TARGET_5TRUE,alpha=1,family = 'binomial',lambda=
donate.cvregpredict = predict(donate.cvregmodel,newx=testmatrix,type='response')


#store the prediction into csv file
logitpredict = as.data.frame(cbind(donate.cvregpredict,testset$ID,testset$TARGDOL))
names(logitpredict)=c('p_logistic','ID','TARGDOL')
write.csv(logitpredict,'~/401-Project1/Data/logitpredict_targetL5_diagonsed_full.csv')

#pickle the model
save(donate.cvregmodel,file='~/401-Project1/Data/logitpredict_targetL10_diagonsed_full.rda')


#accuracy:train
donate.resamplepredict = predict(donate.cvregmodel,newdata=trainmatrix,type='response')
t = table(donate.resamplepredict>0.5,resample$IF_TARGET_10TRUE)
t
sum(diag(t))/sum(t)


#accuracy:test
```

```
#donate.logpredict = predict(donate.logistic,newdata=donate.data[donate.data$IFTEST,c(21,29,32,35,13,12
donate.logpredict = predict(donate.lassomodel,newx=donate.testmatrix,type='response')
t = table(donate.cvregpredict>0.5,testset$IF_TARGET_10TRUE)
t
sum(diag(t))/sum(t)

# top 1000 predicts
ntop=1000
donate.test = donate.data[donate.data$IFTEST,]
testorder=order(-donate.test$TARGDOL)
donate.toptest=donate.test[testorder,]
topid=donate.toptest$ID[1:ntop]

df=as.data.frame(cbind(donate.logpredict,donate.test$TARGDOL,donate.test$ID))
df2=df[order(-df$V2),]
dftop=df2[1:ntop,]


names(df)[c(2,3)]=c('TARDOL','ID')


mean(dftop$s0>.5)

write.csv(df,'~/401-Project1/Data/P_NewLogisticModel.csv')

# load in data
trainset = read.csv('~/401-Project1/Data/trainset_multilinear_noinfluentialnoimputed_standardized.csv')
testset = read.csv('~/401-Project1/Data/testset_standardized.csv')

trainset = trainset[,-1]
# LIST OF VARIABLES
varlist = names(trainset)

# drop responses and IDs
varlist = varlist[-c(1,2,3,4,5,6,23,24,40)]

log_formula = 'TARGDOL~1'
for (i in 1:length(varlist)){
  log_formula = paste(log_formula,varlist[i],sep="+")
}

#add quadratic


for (i in 1:length(varlist)){
  log_formula = paste(log_formula,'+I(',varlist[i],'^2)')
}

#add interaction terms
for (i in 1:(length(varlist)-1)){

  for (j in (i+1):length(varlist)){
    log_formula=paste(log_formula,paste(varlist[i],varlist[j],sep='*'),sep='+')
```

```
  }
}

#formula
log_formula=as.formula(log_formula)


#generate the model as input for glmnet function

trainmatrix = model.matrix(log_formula,data=trainset)

testmatrix = model.matrix(log_formula,data=testset)



# donate.regtrainmatrix = model.matrix(log_formula,data=donate.trainforreg)
cv.regression = cv.glmnet(x=trainmatrix,y=trainset$log_target,alpha=1,family = 'gaussian',nfolds = 5)
plot(cv.regression)
donate.cvregmodel = glmnet(x=trainmatrix,y=trainset$log_target,alpha=1,family = 'gaussian',lambda=cv.reg
donate.cvregpredict = predict(donate.cvregmodel,newx=testmatrix)

#store the prediction into csv file
regpredict = as.data.frame(cbind(donate.cvregpredict,testset$ID,testset$TARGDOL))
write.csv(regpredict,'~/401-Project1/Data/regpredict_diagnosed_fullmodel2.csv')

#pickle the model
save(donate.cvregmodel,file='~/401-Project1/Data/reg_diagnosed_full2.rda')

reg = read.csv('~/401-Project1/Data/regpredict_diagnosed_fullmodel2.csv')

#check MSE
reg$V4<-ifelse(reg$V3==0,1,reg$V3)
reg$V5<-reg$s0*0.39732633+2.232861e-01

sqrt(mean(( reg$V5 - log10(reg$V4) )^2))

sd(log10(reg$V5))

# Check revenue
reg = read.csv('~/401-Project1/Data/regpredict_diagnosed_fullmodel2.csv')
log = read.csv('~/401-Project1/Data/logitpredict_targetL5_diagonsed_full.csv')
predfull = cbind(reg,log)
predfull = cbind(predfull,predfull$s0*predfull$p_logistic)
names(predfull)[9] = 'product_pred'
predfull=predfull[order(-predfull$product_pred),]
sum(predfull$TARGDOL[1:1000])
```