Derek Hughes
PRED 422- Sec55
Spring 2015
Final Project Code

```
# PREDICT 422 Practical Machine Learning

# Course Project - Example R Script File

# OBJECTIVE: A charitable organization wishes to develop a machine learning
# model to improve the cost-effectiveness of their direct marketing campaigns
# to previous donors.

# 1) Develop a classification model using data from the most recent campaign that
# can effectively capture likely donors so that the expected net profit is maximized.

# 2) Develop a prediction model to predict donation amounts for donors - the data
# for this will consist of the records for donors only.

# load the data
charity <- read.csv(file.choose()) # load the "charity.csv" file

dim(charity) # dimensions of charity data set (8009 total over 24 variables)
fix(charity) # view data set

names(charity) # names of variables in charity.t
attach(charity) # enable R to use variable names in coding

summary(charity) # numeric summary of each variable (2007-test, 3984-train, 2018-
valid)
	# only missing values found in damt and donr variables where test set,
appropriately, has them missing
	# five regions represented but only four listed as variables
	# hinc numerically represents 7 different categorical levels of household
income
	# wrat - weath rating based on 9 different segment levels represented
numerically

cor(charity[,-24]) # correlations between variables
# stronger correlations between agif-rgif-lgif, plow-avhv-incm-inca

#pairs(charity) # pairwise plot of variables to look for correlations


# predictor transformations
```

```
# check for normal distribution of each variable also looking for outliers
# by comparing histogram with descriptive stats for each variable

hist(charity$reg1)
hist(charity$reg2)
hist(charity$reg3)
hist(charity$reg4)
hist(charity$home)
hist(charity$chld)
hist(charity$hinc)
hist(charity$genf)
hist(charity$wrat)
hist(charity$avhv)
hist(charity$incm)
hist(charity$inca)
hist(charity$plow)
hist(charity$npro)
hist(charity$tgif)
hist(charity$lgif)
hist(charity$rgif)
hist(charity$tdon)
hist(charity$tlag)
hist(charity$agif)


# data set with transformed variables

charity.t <- charity # charity data set with transformed variables

charity.t$avhv <- log(charity.t$avhv) # log transformation of avhv variable
hist(charity$avhv) # before
hist(charity.t$avhv) # after

charity.t$incm <- log(charity.t$incm) # log transformation of incm variable
hist(charity$incm) # before
hist(charity.t$incm) # after

charity.t$inca <- log(charity.t$inca) # log transformation of inca variable
hist(charity$inca) # before
hist(charity.t$inca) # after

charity.t$plow <- sqrt(charity.t$plow) # sqrt transformation of plow variable
hist(charity$plow) # before
hist(charity.t$plow) # after

charity.t$tgif[charity$tlgif>=600] = 601 # trimming of tgif variable
```

```
charity.t$tgif <- log(charity.t$tgif) # log transformation of tgif variable
hist(charity$tgif) # before
hist(charity.t$tgif) # after

charity.t$lgif[charity$lgif>=250] = 251 # trimming of lgif variable
charity.t$lgif <- log(charity.t$lgif)   # log transformation of lgif variable
hist(charity$lgif) # before
hist(charity.t$lgif) # after

charity.t$rgif <- log(charity.t$rgif) # log transformation of rgif variable
hist(charity$rgif) # before
hist(charity.t$rgif) # after

charity.t$tlag <- log(charity.t$tlag) # log transformation of tlag variable
hist(charity$tlag) # before
hist(charity.t$tlag) # after

charity.t$agif <- log(charity.t$agif) # log transformation of agif variable
hist(charity$agif) # before
hist(charity.t$agif) # after




# set up data for analysis

data.train <- charity.t[charity$part=="train",] # full training set with 24 predictors
x.train <- data.train[,2:21] # removes ID, donr, damt, part from predictor variables, so
only 20 predictors
c.train <- data.train[,22] # donr
n.train.c <- length(c.train) # 3984
y.train <- data.train[c.train==1,23] # damt for observations with donr=1
n.train.y <- length(y.train) # 1995

data.valid <- charity.t[charity$part=="valid",]
x.valid <- data.valid[,2:21] # removes ID, donr, damt, part from predictor variables, so
only 20 predictors
c.valid <- data.valid[,22] # donr
n.valid.c <- length(c.valid) # 2018
y.valid <- data.valid[c.valid==1,23] # damt for observations with donr=1
n.valid.y <- length(y.valid) # 999

data.test <- charity.t[charity$part=="test",]
n.test <- dim(data.test)[1] # 2007
```

```
x.test <- data.test[,2:21] # removes ID, donr, damt, part from predictor variables, so only 20 predictors

x.train.mean <- apply(x.train, 2, mean)
x.train.sd <- apply(x.train, 2, sd)
x.train.std <- t((t(x.train)-x.train.mean)/x.train.sd) # standardize to have zero mean and unit sd
apply(x.train.std, 2, mean) # check zero mean
apply(x.train.std, 2, sd) # check unit sd
data.train.std.c <- data.frame(x.train.std, donr=c.train) # to classify donr
data.train.std.y <- data.frame(x.train.std[c.train==1,], damt=y.train) # to predict damt when donr=1

x.valid.std <- t((t(x.valid)-x.train.mean)/x.train.sd) # standardize using training mean and sd
data.valid.std.c <- data.frame(x.valid.std, donr=c.valid) # to classify donr
data.valid.std.y <- data.frame(x.valid.std[c.valid==1,], damt=y.valid) # to predict damt when donr=1

x.test.std <- t((t(x.test)-x.train.mean)/x.train.sd) # standardize using training mean and sd
data.test.std <- data.frame(x.test.std)



##### VARIABLE SELECTION ######

library(leaps)

# for damt as response
regfit.full=regsubsets(damt~.,data=charity.t[,2:23], nvmax=21)
best.summary=summary(regfit.full)
which.min(best.summary$bic)
coef(regfit.full,13)
# variables selected (13): reg1 + reg2 + reg3 + reg4 + chld + hinc + incm + plow + tgif + lgif + rgif + agif + donr

# for donr as response
regfit.full=regsubsets(donr~.,data=charity.t[,2:22], nvmax=21)
best.summary=summary(regfit.full)
which.min(best.summary$bic)
coef(regfit.full,9)
# variables selected (9): reg1 + reg2 + home + chld + wrat + incm + tgif + tdon + tlag


# for damt as response
regfit.fwd=regsubsets(damt~.,data=charity.t[,2:23], method="forward", nvmax=21)
```

```
fwd.summary=summary(regfit.fwd)
which.min(fwd.summary$bic)
coef(regfit.fwd,13)
# same # and variables selected as best subset (11)

# for donr as response
regfit.fwd=regsubsets(donr~.,data=charity.t[,2:22], method="forward", nvmax=21)
fwd.summary=summary(regfit.fwd)
which.min(fwd.summary$bic)
coef(regfit.fwd,9)
# same # and variables selected as best subset for donr (9)


# for damt as response
regfit.bkwd=regsubsets(damt~.,data=charity.t[,2:23], method="backward", nvmax=21)
bkwd.summary=summary(regfit.bkwd)
which.min(bkwd.summary$bic)
coef(regfit.bkwd,13)
# same # and variables selected as best subset (11)

# for donr as response
regfit.bkwd=regsubsets(donr~.,data=charity.t[,2:22], method="backward", nvmax=21)
bkwd.summary=summary(regfit.bkwd)
which.min(bkwd.summary$bic)
coef(regfit.bkwd,13)
# same # and variables selected as best subset for donr (9)


# other variables selected in addition to the aforementioned 11
# via CV: incm plow
# via p-value in linear regression: reg2 genf tdon home

# total variables selected from various methods: 14
# a model with just these 14 variables tested on the different
# classification and regression models and evaulated on improvements
# in profitability and test MSE




##### CLASSIFICATION MODELING ######

# linear discriminant analysis

library(MASS)
```

```
set.seed(11)
model.lda1 <- lda(donr ~ reg1 + reg2 + reg3 + reg4 + home + chld + hinc + I(hinc^2) +
genf + wrat +
                    avhv + incm + inca + plow + npro + tgif + lgif + rgif + tdon + I(tdon^2) +
tlag + agif,
                  data=data.train.std.c) # include additional terms on the fly using I()

#model.lda2 <- lda(donr ~ reg3 + reg4 + chld + hinc + incm + plow + tgif + lgif + rgif +
agif,
#                   data.train.std.c) # include additional terms on the fly using I()

# test removal of inca and lgif b/c of vif >5 for those predictors
#model.lda3 <- lda(donr ~ reg1 + reg2 + reg3 + reg4 + home + chld + hinc + I(hinc^2) +
genf + wrat +
#                     avhv + incm + plow + npro + tgif + rgif + tdon + tlag + agif,
#                   data.train.std.c)

# using subset of terms found by best subset, forward, and backward stepwise methods
#model.lda4 <- lda(donr~ reg1 + reg2 + home + chld + wrat + incm + tgif + tdon + tlag,
data=data.train.std.c)


coef(model.lda1)
# Note: strictly speaking, LDA should not be used with qualitative predictors,
# but in practice it often is if the goal is simply to find a good predictive model

post.valid.lda1 <- predict(model.lda1, data.valid.std.c)$posterior[,2] # n.valid.c post
probs


# calculate ordered profit function using average donation = $14.50 and mailing cost = $2
profit.lda1 <- cumsum(14.5*c.valid[order(post.valid.lda1, decreasing=T)]-2)

n.mail.valid <- which.max(profit.lda1) # number of mailings that maximizes profits
plot(profit.lda1) # see how profits change as more mailings are made


c(n.mail.valid, max(profit.lda1)) # report number of mailings and maximum profit
# 1356.0 11657.5

cutoff.lda1 <- sort(post.valid.lda1, decreasing=T)[n.mail.valid+1] # set cutoff based on
n.mail.valid
chat.valid.lda1 <- ifelse(post.valid.lda1>cutoff.lda1, 1, 0) # mail to everyone above the
cutoff
table(chat.valid.lda1, c.valid) # classification table
#                c.valid
```

```
#chat.valid.lda1   0   1
#               0 654   8
#               1 365 991
# check n.mail.valid = 365+991 = 1356
# check profit = 14.5*991-2*1329 = 11657.5

# sensitivity = 991/(991+8) = .99199
# specificity = 1-(365/(365+654)) = .64181
# error rate = 373/2018 = .18483


# ROC and AUC
library(pROC)

rocLda1=roc(c.valid, post.valid.lda1)
plot(rocLda1, main="LDA1")
# AUC = 0.9485




# quadratic discriminant analysis
set.seed(11)
model.qda1 <- qda(donr ~ reg1 + reg2 + reg3 + reg4 + home + chld + hinc + I(hinc^2) +
genf + wrat +
                  avhv + incm + inca + plow + npro + tgif + lgif + rgif + tdon + tlag + agif,
                data=data.train.std.c) # include additional terms on the fly using I()

model.qda2 <- qda(donr ~ reg3 + reg4 + chld + hinc + incm + plow + tgif + lgif + rgif +
agif,
                data.train.std.c) # include additional terms on the fly using I()

# test removal of inca and lgif b/c of vif >5 for those predictors
model.qda3 <- qda(donr ~ reg1 + reg2 + reg3 + reg4 + home + chld + hinc + I(hinc^2) +
genf + wrat +
                  avhv + incm + plow + npro + tgif + rgif + tdon + tlag + agif,
                data=data.train.std.c)

# using subset of terms found by best subset, forward, and backward stepwise methods
model.qda4 <- qda(donr~ reg1 + reg2 + home + chld + wrat + incm + tgif + tdon + tlag,
data=data.train.std.c)


# Note: strictly speaking, QDA should not be used with qualitative predictors,
# but in practice it often is if the goal is simply to find a good predictive model
```

```
post.valid.qda3 <- predict(model.qda3, data.valid.std.c)$posterior[,2] # n.valid.c post
probs

# calculate ordered profit function using average donation = $14.50 and mailing cost = $2

profit.qda3 <- cumsum(14.5*c.valid[order(post.valid.qda3, decreasing=T)]-2)
plot(profit.qda3) # see how profits change as more mailings are made
n.mail.valid <- which.max(profit.qda3) # number of mailings that maximizes profits
c(n.mail.valid, max(profit.qda3)) # report number of mailings and maximum profit
# 1423.0 11233.5 - qda1
# 1313   11236 - qda3

cutoff.qda3 <- sort(post.valid.qda3, decreasing=T)[n.mail.valid+1] # set cutoff based on
n.mail.valid
chat.valid.qda3 <- ifelse(post.valid.qda3>cutoff.qda3, 1, 0) # mail to everyone above the
cutoff
table(chat.valid.qda3, c.valid) # classification table
#              c.valid
#chat.valid.qda1   0   1
#             0 567  28
#             1 452 971
# check n.mail.valid = 452+971 = 1423.0
# check profit = 14.5*971-2*1300 = 11233.5

# sensitivity = 971/(971+28) = .97197
# specificity = 1-(452/(452+673)) = .55643
# error rate = 480/2018 = .23786


#chat.valid.qda3    0   1
#              0 662  43
#              1 357 956
# check n.mail.valid = 357+956 = 1313.0
# check profit = 14.5*956-2*1313 = 11236

# sensitivity = 956/(956+43) = .95696
# specificity = 1-(357/(357+662)) = .64966
# error rate = 400/2018 = .19822


# ROC and AUC
rocQda1=roc(c.valid, post.valid.qda3)
plot(rocQda1, main="QDA3")
# AUC = .9166 - qda1
# AUC = .918 - qda3
```

```
# logistic regression
set.seed(11)
model.log1 <- glm(donr ~ reg1 + reg2 + reg3 + reg4 + home + chld + hinc + I(hinc^2) +
genf + wrat +
                    avhv + incm + inca + plow + npro + tgif + lgif + rgif + tdon + I(tdon^2) +
tlag + agif,
                  data=data.train.std.c, family=binomial("logit"))

model.log2 <- glm(donr ~ reg3 + reg4 + chld + hinc + incm + plow + tgif + lgif + rgif +
agif,
                  data.train.std.c, family=binomial("logit")) # include additional terms on the fly
using I()

# using subset of terms found by best subset, forward, and backward stepwise methods
model.log3 <- glm(donr~ reg1 + reg2 + home + chld + wrat + incm + tgif + tdon + tlag,
                  data=data.train.std.c, family=binomial("logit"))


post.valid.log1 <- predict(model.log1, data.valid.std.c, type="response") # n.valid post
probs

# calculate ordered profit function using average donation = $14.50 and mailing cost = $2

profit.log1 <- cumsum(14.5*c.valid[order(post.valid.log1, decreasing=T)]-2)
plot(profit.log1) # see how profits change as more mailings are made
n.mail.valid <- which.max(profit.log1) # number of mailings that maximizes profits
c(n.mail.valid, max(profit.log1)) # report number of mailings and maximum profit
# 1341 11702

cutoff.log1 <- sort(post.valid.log1, decreasing=T)[n.mail.valid+1] # set cutoff based on
n.mail.valid
chat.valid.log1 <- ifelse(post.valid.log1>cutoff.log1, 1, 0) # mail to everyone above the
cutoff
table(chat.valid.log1, c.valid) # classification table
#              c.valid
#chat.valid.log1   0   1
#             0 670   7
#             1 349 992
# check n.mail.valid = 349+992 = 1341
# check profit = 14.5*992-2*1291 = 11702
```

```
# sensitivity = 992/(992+7) = .99299
# specificity = 1-(349/(349+670)) = .65751
# error rate = 356/2018 = .17641


# ROC and AUC
library(pROC)

# ROC
rocLog1=roc(c.valid,post.valid.log1)
plot(rocLog1, main="Log1")
# AUC = .95

coef(model.log1)




# logistic regression GAM

library(gam)
set.seed(11)
model.logGAM1 <- gam(donr ~ reg1 + reg2 + reg3 + reg4 + home + s(chld,4) + s(hinc,4)
+ genf + wrat +
                    avhv + incm + inca + plow + npro + tgif + lgif + rgif + s(tdon,4) + tlag +
agif,
                  data=data.train.std.c, family=binomial)

model.logGAM2 <- gam(donr ~ reg1 + reg2 + reg3 + reg4 + home + s(chld,4) + s(hinc,4)
+ genf + wrat +
                      s(avhv,4) + incm + inca + plow + npro + tgif + lgif + rgif + s(tdon,4) + tlag
+ agif,
                     data=data.train.std.c, family=binomial)

model.logGAM3 <- gam(donr ~ reg1 + reg2 + reg3 + reg4 + home + s(chld,4) + s(hinc,4)
+ genf + wrat +
                      s(avhv,4) + s(incm,4) + inca + plow + npro + tgif + lgif + rgif + s(tdon,4) +
tlag + agif,
                     data=data.train.std.c, family=binomial)

model.logGAM4 <- gam(donr ~ reg1 + reg2 + reg3 + reg4 + home + s(chld,4) + s(hinc,4)
+ genf + wrat +
                     s(avhv,4) + s(incm,4) + s(inca,4) + plow + npro + tgif + lgif + rgif +
s(tdon,4) + tlag + agif,
                     data=data.train.std.c, family=binomial)
```

```
model.logGAM5 <- gam(donr~ reg1 + reg2 + home + s(chld,4) + wrat + s(incm,4) + tgif
+ s(tdon,4) + tlag,
                    data=data.train.std.c, family=binomial)

# anova(model.logGAM1,model.logGAM2,model.logGAM3,model.logGAM4,test="F")
# anova indicates model 2 is most significant

post.valid.logGAM1 <- predict(model.logGAM1, data.valid.std.c, type="response") #
n.valid post probs

# calculate ordered profit function using average donation = $14.50 and mailing cost = $2

profit.logGAM1 <- cumsum(14.5*c.valid[order(post.valid.logGAM1, decreasing=T)]-2)
plot(profit.logGAM1) # see how profits change as more mailings are made
n.mail.valid <- which.max(profit.logGAM1) # number of mailings that maximizes profits
c(n.mail.valid, max(profit.logGAM1)) # report number of mailings and maximum profit
# 1245 11850.5

cutoff.logGAM1 <- sort(post.valid.logGAM1, decreasing=T)[n.mail.valid+1] # set cutoff
based on n.mail.valid
chat.valid.logGAM1 <- ifelse(post.valid.logGAM1>cutoff.logGAM1, 1, 0) # mail to
everyone above the cutoff
table(chat.valid.logGAM1, c.valid) # classification table
#              c.valid
#chat.valid.log1   0   1
#             0 763  10
#             1 256 989
# check n.mail.valid = 256+989 = 1245
# check profit = 14.5*989-2*1245 = 11850.5

# sensitivity = 989/(989+10) = .98999
# specificity = 1-(256/(256+763)) = .74887
# error rate = 266/2018 = .13181

# ROC and AUC
library(pROC)

# ROC curve
rocLogGAM1=roc(c.valid,post.valid.logGAM1)
plot(rocLogGAM1, main="LogGAM1")
# AUC = 0.9663

coef(model.logGAM1)
```

```
# k-nearest neighbors

library(class)

# k=1
set.seed(11)
knn.pred.k1=knn(data.train.std.c, data.valid.std.c, c.train, k=1)
table(knn.pred.k1,c.valid)
mean(knn.pred.k1==c.valid)

#              c.valid
#chat.valid.log1   0   1
#             0 730 158
#             1 289 841
# check n.mail.valid = 289+841 = 1130
# check profit = 14.5*841-2*1130 = 9934.5

# k=5
set.seed(11)
knn.pred.k5=knn(x.train.std, x.valid.std, c.train, k=5)
table(knn.pred.k5,c.valid)
mean(knn.pred.k5==c.valid)

#              c.valid
#chat.valid.log1   0   1
#             0 726  88
#             1 293 911
# check n.mail.valid = 293+911 = 1204
# check profit = 14.5*913-2*1204 = 10830.5

# k=10
set.seed(11)
knn.pred.k10=knn(x.train.std, x.valid.std, c.train, k=10)
table(knn.pred.k10,c.valid)
mean(knn.pred.k10==c.valid)

#              c.valid
#chat.valid.log1   0   1
#             0 701  60
#             1 318 939
# check n.mail.valid = 318+939 = 1257
# check profit = 14.5*939-2*1257 = 11101.5

# k=19
```

```
set.seed(11)
knn.pred.k19=knn(x.train.std, x.valid.std, c.train, k=19)
table(knn.pred.k19,c.valid)
mean(knn.pred.k19==c.valid)
# 0.808226
dim(x.train.std)

#              c.valid
#chat.valid.log1   0   1
#              0 686  54
#              1 333 945
# check n.mail.valid = 333+945 = 1278
# check profit = 14.5*945-2*1278 = 11146.5 - -  *** BEST KNN at K=19 **

# sensitivity = 945/(945+54) = .94595
# specificity = 1-(333/(333+686)) = .67321
# error rate = 333/2018 = .16501


# ROC and AUC
library(pROC)

rocKnn19=roc(c.valid, as.numeric(knn.pred.k19))
plot(rocKnn19,main="KNN 19")
# AUC = 0.8096






# random forest

library(randomForest)

set.seed(11)
model.rf1 <- randomForest(donr ~ reg1 + reg2 + reg3 + reg4 + home + chld + hinc +
I(hinc^2) + genf + wrat +
                   avhv + incm + plow + npro + tgif + rgif + tdon + I(tdon^2) + tlag + agif,
                  data=data.train.std.c, mtry=5, importance=TRUE)
# using sqrt(p=22 variables) for mtry=5 for classification approach

#model.rf2 <- randomForest(donr ~ reg3 + reg4 + chld + hinc + incm + plow + tgif + lgif
+ rgif + agif,
#                 data.train.std.c, mtry=20, importance=TRUE) # include additional terms on
the fly using I()
```

```
#model.rf3 <- randomForest(donr ~ reg1 + reg2 + reg3 + reg4 + home + chld + hinc +
I(hinc^2) + genf + wrat +
#                          avhv + incm + plow + npro + tgif + rgif + tdon + tlag + agif,
#                          data=data.train.std.c, mtry=20, importance=TRUE)

#model.rf4 <- randomForest(donr~ reg1 + reg2 + home + chld + wrat + incm + tgif +
tdon + tlag,
#                         data=data.train.std.c, mtry=9, importance=TRUE)


post.valid.rf1 <- predict(model.rf1, newdata=data.valid.std.c) # n.valid post probs

# calculate ordered profit function using average donation = $14.50 and mailing cost = $2

importance(model.rf1)
# most important variables: chld, hinc^2, reg2, home, wrat

profit.rf1 <- cumsum(14.5*c.valid[order(post.valid.rf1, decreasing=T)]-2)
plot(profit.rf1) # see how profits change as more mailings are made
n.mail.valid <- which.max(profit.rf1) # number of mailings that maximizes profits
c(n.mail.valid, max(profit.rf1)) # report number of mailings and maximum profit
# 1220.0 11784.5

cutoff.rf1 <- sort(post.valid.rf1, decreasing=T)[n.mail.valid+1] # set cutoff based on
n.mail.valid
chat.valid.rf1 <- ifelse(post.valid.rf1>cutoff.rf1, 1, 0) # mail to everyone above the cutoff
table(chat.valid.rf1, c.valid) # classification table
#              c.valid
#chat.valid.log1   0   1
#             0 780  18
#             1 239 981
# check n.mail.valid = 239+981 = 1220
# check profit = 14.5*981-2*1220 = 11784.5

# sensitivity = 981/(981+18) = .981981
# specificity = 1-(239/(239+825)) = .7655
# error rate = 257/2018 = .127354


# ROC and AUC
library(pROC)

# ROC curve
rocRf1=roc(c.valid,post.valid.rf1)
```

```
plot(rocRf1, main="Random Forest 1")
# AUC = 0.965

# boosting

library(gbm)

set.seed(11)
model.boost1 <- gbm(donr ~ reg1 + reg2 + reg3 + reg4 + home + chld + hinc + I(hinc^2)
+ genf + wrat +
                    avhv + incm + inca + plow + npro + tgif + lgif + rgif + tdon + tlag + agif,
                  data=data.train.std.c, distribution="gaussian", n.trees=5000,
interaction.depth=4,shrinkage=0.01)

#model.boost2 <- gbm(donr ~ reg2 + genf + tdon + home + reg3 + reg4 + chld + hinc +
incm + plow + tgif + lgif + rgif + agif,
#                   data=data.train.std.c, distribution="gaussian", n.trees=5000,
interaction.depth=4,shrinkage=0.01)

#model.boost3 <- gbm(donr ~ reg1 + reg2 + reg3 + reg4 + home + chld + hinc +
I(hinc^2) + genf + wrat +
#                    avhv + incm + plow + npro + tgif + rgif + tdon + tlag + agif,
#                  data=data.train.std.c, distribution="gaussian", n.trees=5000,
interaction.depth=4,shrinkage=0.01)


post.valid.boost1 <- predict(model.boost1, newdata=data.valid.std.c, n.trees=5000) #
n.valid post probs

# calculate ordered profit function using average donation = $14.50 and mailing cost = $2

profit.boost1 <- cumsum(14.5*c.valid[order(post.valid.boost1, decreasing=T)]-2)
plot(profit.boost1) # see how profits change as more mailings are made
n.mail.valid <- which.max(profit.boost1) # number of mailings that maximizes profits
c(n.mail.valid, max(profit.boost1)) # report number of mailings and maximum profit
# 1188 11863

cutoff.boost1 <- sort(post.valid.boost1, decreasing=T)[n.mail.valid+1] # set cutoff based
on n.mail.valid
chat.valid.boost1 <- ifelse(post.valid.boost1>cutoff.boost1, 1, 0) # mail to everyone
above the cutoff
```

```
table(chat.valid.boost1, c.valid) # classification table
#              c.valid
#chat.valid.log1   0   1
#             0 813  17
#             1 206 982
# check n.mail.valid = 206+982 = 1188
# check profit = 14.5*982-2*1188 = 11863

# sensitivity = 982/(982+17) = .983
# specificity = 1-(206/(206+813)) = .7978
# error rate = 206/2018 = .1021


# ROC and AUC
library(pROC)

# ROC curve
rocBoost1=roc(c.valid,post.valid.boost1)
plot(rocBoost1, main="Boosting 1")
# AUC = 0.9672






# support vector machines

library(e1071)

set.seed(11)
model.svm1 <- tune(svm, donr ~ reg1 + reg2 + reg3 + reg4 + home + chld + hinc +
I(hinc^2) + genf + wrat +
                  avhv + incm + inca + plow + npro + tgif + lgif + rgif + tdon + I(tdon^2) +
tlag + agif,
                data=data.train.std.c, kernel="linear", ranges=list(cost=c(.001)))

bestmod=model.svm1$best.model
bestmod
post.valid.svm1=predict(bestmod,data.valid.std.c)

#post.valid.svm1 <- predict(model.svm1, newdata=data.valid.std.c, n.trees=5000) #
n.valid post probs

# calculate ordered profit function using average donation = $14.50 and mailing cost = $2
```

```
profit.svm1 <- cumsum(14.5*c.valid[order(post.valid.svm1, decreasing=T)]-2)
plot(profit.svm1) # see how profits change as more mailings are made
n.mail.valid <- which.max(profit.svm1) # number of mailings that maximizes profits
c(n.mail.valid, max(profit.svm1)) # report number of mailings and maximum profit

# 1374 11636 c=.001
# 1351 11595 c=.01
# 1356 11585 c=.1
# 1369 11588 c=10

cutoff.svm1 <- sort(post.valid.svm1, decreasing=T)[n.mail.valid+1] # set cutoff based on
n.mail.valid
chat.valid.svm1 <- ifelse(post.valid.svm1>cutoff.svm1, 1, 0) # mail to everyone above
the cutoff
table(chat.valid.svm1, c.valid) # classification table
#              c.valid
#chat.valid.log1   0   1
#             0 637   7
#             1 382 992
# check n.mail.valid = 382+992 = 1374
# check profit = 14.5*992-2*1188 = 11636

# sensitivity = 992/(992+7) = .99299
# specificity = 1-(382/(382+637)) = .6251
# error rate = 382/2018 = .1893

# ROC and AUC
library(pROC)

# ROC curve
rocSvm1=roc(c.valid,post.valid.svm1)
plot(rocSvm1, main="SVM 1")
# AUC = 0.9467


# Results

# n.mail Profit  Model
# 1188   11863   Boost1
# 1245   11850.5 LogGAM1
# 1220   11784.5 RF1
```

```
# 1341   11702   Log1
# 1356   11657.5 LDA1
# 1374   11636   SVM1
# 1313   11236   QDA1
# 1278   11146.5 KNN19


# select model.log1 since it has maximum profit in the validation sample

post.test <- predict(model.boost1, data.test.std, type="response", n.trees=5000) # post
probs for test data
```

```
# Oversampling adjustment for calculating number of mailings for test set

n.mail.valid <- which.max(profit.boost1)
tr.rate <- .1 # typical response rate is .1
vr.rate <- .5 # whereas validation response rate is .5
adj.test.1 <- (n.mail.valid/n.valid.c)/(vr.rate/tr.rate) # adjustment for mail yes
adj.test.0 <- ((n.valid.c-n.mail.valid)/n.valid.c)/((1-vr.rate)/(1-tr.rate)) # adjustment for
mail no
adj.test <- adj.test.1/(adj.test.1+adj.test.0) # scale into a proportion
n.mail.test <- round(n.test*adj.test, 0) # calculate number of mailings for test set

cutoff.test <- sort(post.test, decreasing=T)[n.mail.test+1] # set cutoff based on n.mail.test
chat.test <- ifelse(post.test>cutoff.test, 1, 0) # mail to everyone above the cutoff
table(chat.test)
#    0    1
# 1731  276
# based on this model we'll mail to the 276 highest posterior probabilities

# See below for saving chat.test into a file for submission
```

PREDICTION MODELING

```
# Least squares regression
set.seed(11)
model.ls1 <- lm(damt ~ reg1 + reg2 + reg3 + reg4 + home + chld + hinc + I(hinc^2) +
genf + wrat +
          avhv + incm + inca + plow + npro + tgif + lgif + rgif + tdon + I(tdon^2) + tlag
+ agif,
        data.train.std.y)

# inca and lgif removed due to vif values > 5
#model.ls2 <- lm(damt ~ reg1 + reg2 + reg3 + reg4 + home + chld + hinc + genf + wrat +
#           avhv + incm+ plow + npro + tgif + rgif + tdon + tlag + agif,
#          data.train.std.y)

# checking vif values on linear model to determine any high multicollinearity > 5
library(usdm)
library(car)
df=model.ls1
vif(df)
# two variables: inca and lgif had vif just over 5 so models with them removed were
tested for any improvements


#model.ls2 <- lm(damt ~ reg2 + genf + tdon + home + reg3 + reg4 + chld + hinc + incm
+ plow + tgif + lgif + rgif + agif,
#               data.train.std.y) # include additional terms on the fly using I()

summary(model.ls1)

pred.valid.ls1 <- predict(model.ls1, newdata = data.valid.std.y) # validation predictions
plot(pred.valid.ls1)

mean((y.valid - pred.valid.ls1)^2) # mean prediction error
# 1.591109
sd((y.valid - pred.valid.ls1)^2)/sqrt(n.valid.y) # std error
# 0.1610636
```

```
coef(model.ls1)




# best subset selection w/ k-fold CV

library(leaps)

# predict function to handle no regsubsets() method
predict.regsubsets <- function(object, newdata, id,...){
  form <- as.formula(object$call[[2]])
  mat <- model.matrix(form, newdata)
  coefi <- coef(object, id = id)
  xvars <- names(coefi)
  mat[, xvars]%*%coefi
}

## TRAINING

# apply 10-fold cross-validation method to select subset of variables
k=10 # list number of folds to create
set.seed(11)
folds=sample(1:k,nrow(data.train.std.y),replace=TRUE)  # create folds using training
data with replacement
cv.errors= matrix(NA,k,10, dimnames=list(NULL,paste(1:10))) # matrix used to store
test errors on subsets


# loop to perform cross-validation process for each fold
# creates 9 training folds and one test fold each iteration
# identifies best subset selection for 9 training folds and then makes prediction on
# the hold out or test fold
# lastly it stores errors in matrix and we calculate MSE for test
for(j in 1:k){
  best.fit=regsubsets(damt~ reg1 + reg2 + reg3 + reg4 + home + chld + hinc + I(hinc^2) +
genf + wrat +
                        avhv + incm + inca + plow + npro + tgif + lgif + rgif + tdon + I(tdon^2) +
tlag + agif,data=data.train.std.y[folds!=j,],nvmax=22)
  for(i in 1:10){
    pred=predict(best.fit,data.train.std.y[folds==j,],id=i)
    cv.errors[j,i]=mean((data.train.std.y$damt[folds==j]-pred)^2)
  }
}
```

```
# find avg MSE value for each model that has a different # of variables for each of 10-cv
folds
mean.cv.errors=apply(cv.errors,2,mean)
which.min(mean.cv.errors)  # we see lowest MSE for 10 fold CV is for 10-variable
model

plot(mean.cv.errors, xlab = "Subset Size", ylab = "CV Errors", pch = 10, type = "l")
points(10, mean.cv.errors[which.min(mean.cv.errors)], pch = 4, col = "red", lwd = 7)

### VALIDATION TESTING

x.all <- charity.t[,2:21]
c.all <- charity.t[,22]
y.all <- charity.t[c.all==1,23]
x.all.mean <- apply(x.all, 2, mean)
x.all.sd <- apply(x.all, 2, sd)
x.all.std <- t((t(x.all)-x.all.mean)/x.all.sd)
data.all.std.y <- data.frame(x.all.std[c.all==1,], damt=y.all) # to predict damt when
donr=1


# apply 10-variable model to full data
set.seed(11)
regfit.best=regsubsets(damt~reg1 + reg2 + reg3 + reg4 + home + chld + hinc + I(hinc^2)
+ genf + wrat +
                     avhv + incm + inca + plow + npro + tgif + lgif + rgif + tdon + I(tdon^2) +
tlag + agif,data=data.all.std.y,nvmax=22)
summary(regfit.best)

# coefficients of best ten variable model on full data set
coef(regfit.best,10)

# apply model from full data set to test data set
test.mat=model.matrix(damt~reg1 + reg2 + reg3 + reg4 + home + chld + hinc + I(hinc^2)
+ genf + wrat +
                    avhv + incm + inca + plow + npro + tgif + lgif + rgif + tdon + I(tdon^2) +
tlag + agif, data=data.valid.std.y) # create matrix for test data

# vector for test MSE values for # variables
val.errors=rep(NA,10)

# extract coefficients from regfit.best for best model of that size & find MSE
coefi=coef(regfit.best,10)
coefi
pred=test.mat[,names(coefi)]%*%coefi
```

```
# determine MSE on test data
val.errors.cv=mean((y.valid-pred)^2)
# show MSE values
val.errors.cv
# 1.608761

# determine SE for MSE
val.errors.cv.se=sd((y.valid-pred)^2)/sqrt(n.valid.y)
# show SE for MSE
val.errors.cv.se
# 0.1643959
```

```
# principal components regression

library(pls)

set.seed(11)
model.pcr1 <- pcr(damt ~ reg1 + reg2 + reg3 + reg4 + home + chld + hinc + I(hinc^2) + genf + wrat +
                  avhv + incm + inca + plow + npro + tgif + lgif + rgif + tdon + I(tdon^2) + tlag + agif,
                data=data.train.std.y, scale=TRUE, validation="CV")

#model.pcr2 <- pcr(damt ~ reg2 + genf + tdon + home + reg3 + reg4 + chld + hinc + incm + plow + tgif + lgif + rgif + agif,
#               data=data.train.std.y, scale=TRUE, validation="CV") # include additional terms on the fly using I()

summary(model.pcr1)
# shows lowest adjCV at 19 comps (identical to comp 21 and 22)

coef(model.pcr1)
par(mfrow = c(1,2))
validationplot(model.pcr1,val.type="MSEP")
# plot of components on MSEP
```

```
pred.valid.pcr1 <- predict(model.pcr1, newdata = data.valid.std.y, ncomp=19) #
validation predictions
plot(pred.valid.pcr1)

mean((y.valid - pred.valid.pcr1)^2) # mean prediction error
# 1.591109-pcr1  #1.607156-pcr2
sd((y.valid - pred.valid.pcr1)^2)/sqrt(n.valid.y) # std error
# 0.1610636-pcr1  #0.1611775-pcr2
```

```
# partial least squares

library(pls)

set.seed(11)
model.pls1 <- plsr(damt ~ reg1 + reg2 + reg3 + reg4 + home + chld + hinc + I(hinc^2) +
genf + wrat +
                avhv + incm + inca + plow + npro + tgif + lgif + rgif + tdon + I(tdon^2) +
tlag + agif,
               data=data.train.std.y, scale=TRUE, validation="CV")

summary(model.pls1)
# 7 components has smallest adjCV

validationplot(model.pls1,val.type="MSEP")

pred.valid.pls1 <- predict(model.pls1, newdata = data.valid.std.y, ncomp=7) # validation
predictions
plot(pred.valid.pls1)

mean((y.valid - pred.valid.pls1)^2) # mean prediction error
# 1.593234
sd((y.valid - pred.valid.pls1)^2)/sqrt(n.valid.y) # std error
# 0.1614317

coef(model.pls1)
```

```
# ridge regression
library(glmnet)

# Ridge regression model using 10-fold cross-validation
x.train.RL <- model.matrix(damt ~ reg1 + reg2 + reg3 + reg4 + home + chld + hinc +
I(hinc^2) + genf + wrat +
                    avhv + incm + inca + plow + npro + tgif + lgif + rgif + tdon + I(tdon^2)
+ tlag + agif,
                    data = data.train.std.y)[,-1]

x.valid.RL <- model.matrix(damt ~ reg1 + reg2 + reg3 + reg4 + home + chld + hinc +
I(hinc^2) + genf + wrat +
                     avhv + incm + inca + plow + npro + tgif + lgif + rgif + tdon +
I(tdon^2) + tlag + agif,
                    data = data.valid.std.y)[,-1]
par(mfrow = c(1,2))
grid <- 10^seq(10, -2, length = 100)
ridge.mod <- glmnet(x.train.RL, y.train, alpha = 0, lambda = grid, thresh = 1e-12)
plot(ridge.mod, xvar = "lambda", label = TRUE)

set.seed(11)
cv.out <- cv.glmnet(x.train.RL, y.train, alpha = 0)
plot(cv.out)

# select best (minimum) lambda value
bestlam=cv.out$lambda.min
bestlam   # min lambda value was 0.125775

ridge.mod <- glmnet(x.train.RL, y.train, alpha = 0, lambda = bestlam)
ridge.pred <- predict(ridge.mod, s = bestlam, newx = x.valid.RL)

mean((ridge.pred - y.valid)^2)
# 1.601929
sd((ridge.pred - y.valid)^2)/sqrt(n.valid.y)
# 0.1624482
```

```
coef(ridge.mod)




# lasso
x.train.las <- model.matrix(damt ~ reg1 + reg2 + reg3 + reg4 + home + chld + hinc +
I(hinc^2) + genf + wrat +
                              avhv + incm + inca + plow + npro + tgif + lgif + rgif + tdon +
I(tdon^2) + tlag + agif,
                            data = data.train.std.y)[,-1]

x.valid.las <- model.matrix(damt ~ reg1 + reg2 + reg3 + reg4 + home + chld + hinc +
I(hinc^2) + genf + wrat +
                              avhv + incm + inca + plow + npro + tgif + lgif + rgif + tdon +
I(tdon^2) + tlag + agif,
                           data = data.valid.std.y)[,-1]

par(mfrow = c(1,2))
grid <- 10^seq(10, -2, length = 100)

lasso.mod <- glmnet(x.train.las, y.train, alpha = 1, lambda = grid, thresh = 1e-12)
plot(lasso.mod, xvar = "lambda", label = TRUE)

set.seed(11)
cv.out <- cv.glmnet(x.train.las, y.train, alpha = 1)
plot(cv.out)

# select best (minimum) lambda value
bestlam=cv.out$lambda.min
bestlam   # min lambda value was 0.004314673

lasso.mod <- glmnet(x.train.las, y.train, alpha = 1, lambda = bestlam)
lasso.pred <- predict(lasso.mod, s = bestlam, newx = x.valid.las)

mean((lasso.pred - y.valid)^2)
# 1.592249
sd((lasso.pred - y.valid)^2)/sqrt(n.valid.y)
# 0.1609076

coef(lasso.mod)
```

```
# random forests

library(gbm)

set.seed(11)
model.rf1 <- randomForest(damt ~ reg1 + reg2 + reg3 + reg4 + home + chld + hinc + I(hinc^2) + genf + wrat +
                   avhv + incm + inca + plow + npro + tgif + lgif + rgif + tdon + I(tdon^2) + tlag + agif,
                   data=data.train.std.y, mtry=7, importance=TRUE)
# using p/3 for mtry=7 for regression approach

#model.rf2 <- randomForest(damt ~ reg2 + genf + tdon + home + reg3 + reg4 + chld + hinc + incm + plow + tgif + lgif + rgif + agif,
#                          data=data.train.std.y, mtry=20, importance=TRUE)

pred.valid.rf1 <- predict(model.rf1, newdata=data.valid.std.y) # n.valid post probs


# validation predictions
plot(pred.valid.rf1)


importance(model.rf1)

mean((pred.valid.rf1-y.valid)^2) # mean prediction error
# 1.661631
sd((pred.valid.rf1-y.valid)^2)/sqrt(n.valid.y) # std error
# 0.1732228

coef(model.rf1)







# boosting
```

```
library(gbm)

set.seed(11)
model.boost1r <- gbm(damt ~ reg1 + reg2 + reg3 + reg4 + home + chld + hinc +
I(hinc^2) + genf + wrat +
                     avhv + incm + inca + plow + npro + tgif + lgif + rgif + tdon + I(tdon^2) +
tlag + agif,
                   data=data.train.std.y, distribution="gaussian", n.trees=5000,
interaction.depth=4,shrinkage=0.01)

#model.boost2r <- gbm(damt ~ reg2 + genf + tdon + home + reg3 + reg4 + chld + hinc +
incm + plow + tgif + lgif + rgif + agif,
#                    data=data.train.std.y, distribution="gaussian", n.trees=5000,
interaction.depth=4,shrinkage=0.01)


# validation predictions
pred.valid.boost1r <- predict(model.boost1r, newdata=data.valid.std.y, n.trees=5000) #
n.valid post probs

plot(pred.valid.boost1r)

mean((pred.valid.boost1r-y.valid)^2) # mean prediction error
# 1.447108
sd((pred.valid.boost1r-y.valid)^2)/sqrt(n.valid.y) # std error
# 0.1660618






# Results

# MPE  Model
# 1.661631 RF1
# 1.608761 BSS1
# 1.601929 RR1
# 1.593234 PLS1
# 1.592249 LAS1
# 1.591109 LS1
# 1.591109 PCR1 *same as LS1 b/c uses all 20 variables
# 1.447108 Boost1
```

select model.ls2 since it has minimum mean prediction error in the validation sample

```
yhat.test <- predict(model.boost1r, newdata = data.test.std, n.trees=5000) # test predictions
```

FINAL RESULTS

Save final results for both classification and regression

```
length(chat.test) # check length = 2007
length(yhat.test) # check length = 2007
chat.test[1:10] # check this consists of 0s and 1s
yhat.test[1:30] # check this consists of plausible predictions of damt

ip <- data.frame(chat=chat.test, yhat=yhat.test) # data frame with two variables: chat and yhat
write.csv(ip, file="DBH.csv", row.names=FALSE) # use your initials for the file name
```

submit the csv file in Angel for evaluation based on actual test donr and damt values