Predict422-CharityProject Part 2

Artur Mrozowski

May 7, 2017

1. Import Data

a. Read the data into R from the CSV file

```
library(leaps)
```

```
## Warning: package 'leaps' was built under R version 3.2.5
```

```
library(glmnet)
```

```
## Warning: package 'glmnet' was built under R version 3.2.5

## Loading required package: Matrix

## Loading required package: foreach

## Warning: package 'foreach' was built under R version 3.2.3

## Loaded glmnet 2.0-5
```

```
library(pls)
```

```
## Warning: package 'pls' was built under R version 3.2.3

##
## Attaching package: 'pls'

## The following object is masked from 'package:stats':
##
##     loadings
```

```
library(car)
```

```
## Warning: package 'car' was built under R version 3.2.3
```

```
charityData = read.csv(file.choose(),na.strings=c("NA"," "))
```

2. Data Preparation Convert categorical variables to factors

The lm() method in R can handle a factor variable without us needing to convert the factor to binary dummy variable(s).

```
charityData$DONR = as.factor(charityData$DONR)
charityData$HOME = as.factor(charityData$HOME)
charityData$HINC = as.factor(charityData$HINC)
```

Subset to observations such that DAMT > 0 (and DONR = 1).

```
regrData = charityData[charityData$DONR == "1",]
rm(charityData)
```

Check for missing Values

```
which(sapply(regrData,anyNA))
```

```
##   HOME   HINC GENDER
##      5      6      7
```

HOME - Make a level 0 and code missing values as 0

```
levels(regrData$HOME) = c(levels(regrData$HOME),"0")
regrData$HOME[is.na(regrData$HOME)] = "0"
table(regrData$HOME,useNA="ifany")
```

```
##
##    0    1
## 1192 2445
```

HINC - Make a level 0 and code missing values as 0

```
levels(regrData$HINC) = c(levels(regrData$HINC),"0")
regrData$HINC[is.na(regrData$HINC)] = "0"
table(regrData$HINC,useNA="ifany")
```

```
##
##   1   2   3   4   5   6   7   0
## 291 525 361 555 701 388 375 441
```

GENDER - Assign A, J, and NA to category U

```
idxMF = regrData$GENDER %in% c("M","F")
regrData$GENDER[!idxMF] = "U"
regrData$GENDER = factor(regrData$GENDER)
table(regrData$GENDER,useNA="ifany")
```

```
##
##    F    M    U
## 1954 1557  126
```

Part C - Re-categorize Variables

Separate RFA Values (R = recency, F = frequency, A = amount)

```
separateRFA = function(xData,varName)
{
  bytes = c("R","F","A")
  newVarNames = paste(varName,bytes, sep="_")

  for (ii in 1:length(bytes)) # Loop over 1 to 3 (corresponding to R, F, and A)
  {
    # Find the unique values for current byte
    byteVals = unique(substr(levels(xData[,varName]),ii,ii))

    for (jj in 1:length(byteVals)) # Loop over unique byte values
    {
      rowIdx = substr(xData[,varName],ii,ii) == byteVals[jj]
      xData[rowIdx,newVarNames[ii]] = byteVals[jj]
    }

    xData[,newVarNames[ii]] = factor(xData[,newVarNames[ii]])
  }

  return(xData)
}
```

```
regrData = separateRFA(regrData,"RFA_96")
#table(regrData$RFA_96,regrData$RFA_96_R)
#table(regrData$RFA_96,regrData$RFA_96_F)
#table(regrData$RFA_96,regrData$RFA_96_A)
```

Drop the variables indicated by dropIdx.

```
dropIdx = which(names(regrData) %in% c("DONR","RFA_96"))


regrData2 = regrData[,-dropIdx]
names(regrData2)
```

```
##  [1] "ID"        "DAMT"      "AGE"       "HOME"      "HINC"      "GENDER"
##  [7] "MEDAGE"    "MEDPPH"    "MEDHVAL"   "MEDINC"    "MEDEDUC"   "NUMPROM"
## [13] "NUMPRM12" "RAMNTALL" "NGIFTALL" "MAXRAMNT" "LASTGIFT" "TDON"
## [19] "RFA_96_R" "RFA_96_F" "RFA_96_A"
```

3. Dataset Partitioning For this assignment, you will employ a hold-out test dataset for model validation and selection.

a. Hold-Out Test Set

b. Training Set 75%

```
# Specify the fraction of data to use in the hold-out test.
testFraction = 0.25
set.seed(123)
```

```
# Sample training subset indices.
# - the index vector has length equal to the size of the sampled set
# - the index values are integer, representing the row numbers to use for the sample
trainIdx = sample(nrow(regrData2),size=(1-testFraction)*nrow(regrData2),
                  replace=FALSE)
```

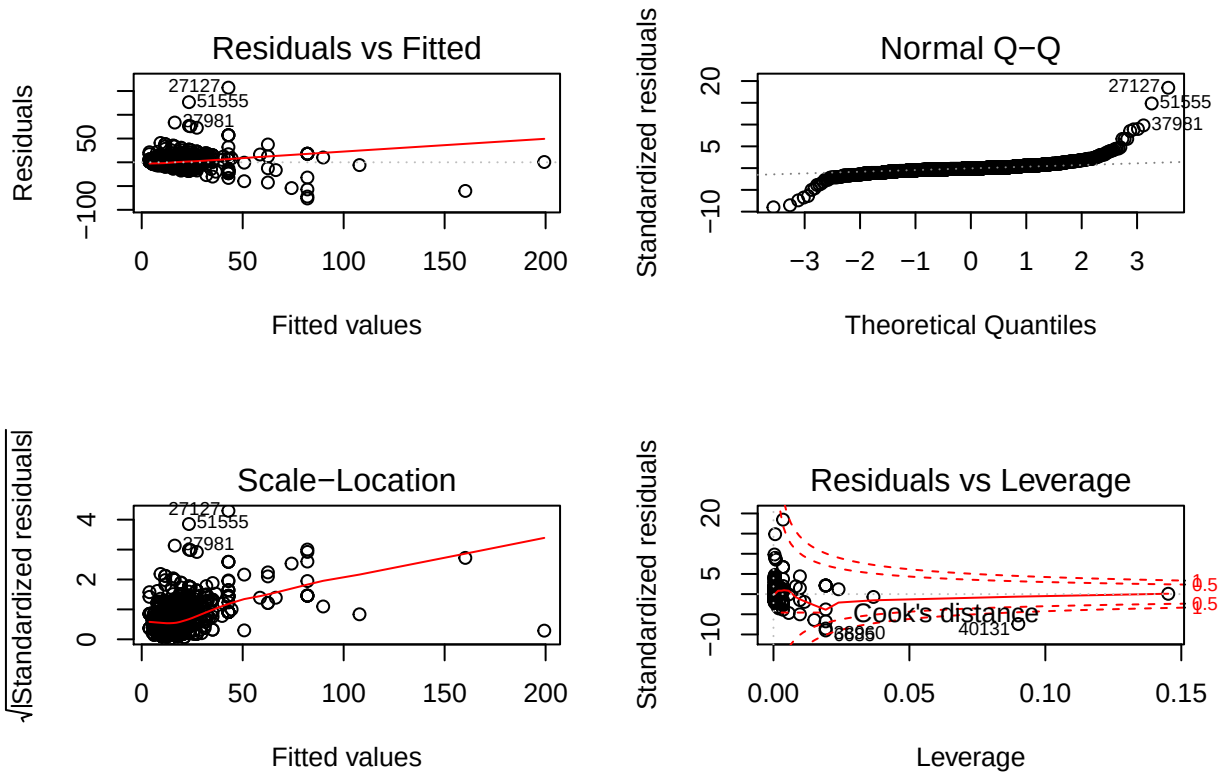
4. Model Fitting

a. Simple linear regression (ISLR Section 3.1) [Recall that simple linear regression is regression with a single predictor variable.]

```
modelA1 = lm(DAMT ~ LASTGIFT,data=regrData2,subset=trainIdx)
summary(modelA1)
```

```
##
## Call:
## lm(formula = DAMT ~ LASTGIFT, data = regrData2, subset = trainIdx)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -76.037  -2.745  -0.566   1.690 157.064
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.83538    0.26786   14.32   <2e-16 ***
## LASTGIFT     0.78202    0.01383   56.55   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.525 on 2725 degrees of freedom
## Multiple R-squared:   0.54,  Adjusted R-squared:  0.5398
## F-statistic:  3198 on 1 and 2725 DF,  p-value: < 2.2e-16
```

```
par(mfrow=c(2,2))
plot(modelA1)
```




```
par(mfrow=c(1,1))
```

b. Multiple linear regression with subset selection (ISLR Section 6.1) Full Regression Model

```
modelB1 = lm(DAMT ~ .-ID,data=regrData2,subset=trainIdx)
summary(modelB1)
```

```
##
## Call:
## lm(formula = DAMT ~ . - ID, data = regrData2, subset = trainIdx)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -62.476  -2.241  -0.432   1.945 155.173
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.973e+00  8.536e+00   0.817 0.414105
## AGE          7.708e-03  1.112e-02   0.693 0.488220
## HOME1       -1.176e+00  4.241e-01  -2.774 0.005577 **
## HINC2       -7.656e-01  6.973e-01  -1.098 0.272296
## HINC3       -3.964e-01  7.608e-01  -0.521 0.602380
## HINC4        4.467e-02  7.230e-01   0.062 0.950746
## HINC5        1.784e-01  7.069e-01   0.252 0.800780
## HINC6       -8.035e-01  7.950e-01  -1.011 0.312260
```

```
## HINC7       -3.360e-01  8.158e-01  -0.412 0.680426
## HINC0       -8.855e-01  7.616e-01  -1.163 0.245015
## GENDERM      1.028e-01  3.255e-01   0.316 0.752167
## GENDERU      7.498e-01  9.192e-01   0.816 0.414722
## MEDAGE      -3.111e-03  2.048e-02  -0.152 0.879260
## MEDPPH      -5.019e-03  3.722e-03  -1.349 0.177568
## MEDHVAL     -3.321e-05  2.348e-04  -0.141 0.887539
## MEDINC       1.963e-03  1.649e-03   1.190 0.233989
## MEDEDUC     -2.802e-03  1.225e-02  -0.229 0.819121
## NUMPROM     -1.463e-02  1.623e-02  -0.902 0.367269
## NUMPRM12     1.487e-02  4.908e-02   0.303 0.761919
## RAMNTALL     1.128e-02  2.705e-03   4.171 3.12e-05 ***
## NGIFTALL    -1.304e-01  3.947e-02  -3.304 0.000966 ***
## MAXRAMNT     6.417e-02  2.597e-02   2.471 0.013545 *
## LASTGIFT     5.230e-01  2.849e-02  18.358  < 2e-16 ***
## TDON         6.894e-02  5.023e-02   1.372 0.170040
## RFA_96_RF    3.724e-01  8.510e-01   0.438 0.661753
## RFA_96_RL    2.356e+00  2.308e+00   1.021 0.307474
## RFA_96_RN   -3.166e-01  8.037e-01  -0.394 0.693711
## RFA_96_RS   -3.248e-02  4.799e-01  -0.068 0.946039
## RFA_96_F2   -7.756e-01  4.799e-01  -1.616 0.106138
## RFA_96_F3   -1.662e+00  5.575e-01  -2.980 0.002905 **
## RFA_96_F4   -1.591e+00  6.223e-01  -2.556 0.010631 *
## RFA_96_AC   -2.870e+00  1.009e+01  -0.284 0.776109
## RFA_96_AD   -7.207e-01  8.255e+00  -0.087 0.930443
## RFA_96_AE    1.107e-01  8.268e+00   0.013 0.989314
## RFA_96_AF    1.464e+00  8.281e+00   0.177 0.859662
## RFA_96_AG    5.708e+00  8.305e+00   0.687 0.491963
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.191 on 2691 degrees of freedom
## Multiple R-squared:  0.5806, Adjusted R-squared:  0.5751
## F-statistic: 106.4 on 35 and 2691 DF,  p-value: < 2.2e-16
```

Checking collinearility. Less than 10 so it seems ok.

```
vif(modelB1)
```

```
##              GVIF Df GVIF^(1/(2*Df))
## AGE      1.226006  1        1.107251
## HOME     1.604210  1        1.266574
## HINC     2.360265  7        1.063261
## GENDER   1.114771  2        1.027534
## MEDAGE   1.165619  1        1.079638
## MEDPPH   1.363546  1        1.167710
## MEDHVAL  2.409119  1        1.552134
## MEDINC   3.423275  1        1.850209
## MEDEDUC  1.805873  1        1.343828
## NUMPROM  5.621034  1        2.370872
## NUMPRM12 2.521013  1        1.587770
## RAMNTALL 3.943013  1        1.985702
## NGIFTALL 5.608374  1        2.368201
```

```
## MAXRAMNT 5.370832  1        2.317505
## LASTGIFT 4.598056  1        2.144308
## TDON     1.815676  1        1.347470
## RFA_96_R 3.176464  4        1.155428
## RFA_96_F 2.754789  3        1.183990
## RFA_96_A 3.672300  5        1.138922
```

```
regfit.fwd=regsubsets (DAMT ~ .-ID,data=regrData2[trainIdx,] ,nvmax =20,method ="forward")
```

```
summary(regfit.fwd)
```

```
## Subset selection object
## Call: regsubsets.formula(DAMT ~ . - ID, data = regrData2[trainIdx,
##     ], nvmax = 20, method = "forward")
## 35 Variables  (and intercept)
##           Forced in Forced out
## AGE           FALSE      FALSE
## HOME1         FALSE      FALSE
## HINC2         FALSE      FALSE
## HINC3         FALSE      FALSE
## HINC4         FALSE      FALSE
## HINC5         FALSE      FALSE
## HINC6         FALSE      FALSE
## HINC7         FALSE      FALSE
## HINC0         FALSE      FALSE
## GENDERM       FALSE      FALSE
## GENDERU       FALSE      FALSE
## MEDAGE        FALSE      FALSE
## MEDPPH        FALSE      FALSE
## MEDHVAL       FALSE      FALSE
## MEDINC        FALSE      FALSE
## MEDEDUC       FALSE      FALSE
## NUMPROM       FALSE      FALSE
## NUMPRM12      FALSE      FALSE
## RAMNTALL      FALSE      FALSE
## NGIFTALL      FALSE      FALSE
## MAXRAMNT      FALSE      FALSE
## LASTGIFT      FALSE      FALSE
## TDON          FALSE      FALSE
## RFA_96_RF     FALSE      FALSE
## RFA_96_RL     FALSE      FALSE
## RFA_96_RN     FALSE      FALSE
## RFA_96_RS     FALSE      FALSE
## RFA_96_F2     FALSE      FALSE
## RFA_96_F3     FALSE      FALSE
## RFA_96_F4     FALSE      FALSE
## RFA_96_AC     FALSE      FALSE
## RFA_96_AD     FALSE      FALSE
## RFA_96_AE     FALSE      FALSE
## RFA_96_AF     FALSE      FALSE
## RFA_96_AG     FALSE      FALSE
## 1 subsets of each size up to 20
## Selection Algorithm: forward
```

```
##           AGE HOME1 HINC2 HINC3 HINC4 HINC5 HINC6 HINC7 HINC0 GENDERM
## 1  ( 1 )  " " " " " "   " "   " "   " "   " "   " "   " "   " "   " "
## 2  ( 1 )  " " " " " "   " "   " "   " "   " "   " "   " "   " "   " "
## 3  ( 1 )  " " " " " "   " "   " "   " "   " "   " "   " "   " "   " "
## 4  ( 1 )  " " " " " "   " "   " "   " "   " "   " "   " "   " "   " "
## 5  ( 1 )  " " " " " "   " "   " "   " "   " "   " "   " "   " "   " "
## 6  ( 1 )  " " " " " "   " "   " "   " "   " "   " "   " "   " "   " "
## 7  ( 1 )  " " " " " "   " "   " "   " "   " "   " "   " "   " "   " "
## 8  ( 1 )  " " " "*"   " "   " "   " "   " "   " "   " "   " "   " "
## 9  ( 1 )  " " " "*"   " "   " "   " "   " "   " "   " "   " "   " "
## 10  ( 1 ) " " " "*"   " "   " "   " "   " "   " "   " "   " "   " "
## 11  ( 1 ) " " " "*"   " "   " "   " "   " "   " "   " "   " "   " "
## 12  ( 1 ) " " " "*"   " "   " "   " "   "*"   " "   " "   " "   " "
## 13  ( 1 ) " " " "*"   " "   " "   " "   "*"   " "   " "   " "   " "
## 14  ( 1 ) " " " "*"   " "   " "   "*"   "*"   " "   " "   " "   " "
## 15  ( 1 ) " " " "*"   " "   " "   "*"   "*"   " "   " "   " "   " "
## 16  ( 1 ) " " " "*"   " "   " "   "*"   "*"   " "   " "   " "   " "
## 17  ( 1 ) " " " "*"   " "   " "   "*"   "*"   " "   " "   " "   " "
## 18  ( 1 ) " " " "*"   " "   " "   "*"   "*"   " "   " "   " "   " "
## 19  ( 1 ) "*" "*"   " "   " "   "*"   "*"   " "   " "   " "   " "
## 20  ( 1 ) "*" "*"   "*"   " "   "*"   "*"   " "   " "   " "   " "
##           GENDERU MEDAGE MEDPPH MEDHVAL MEDINC MEDEDUC NUMPROM NUMPRM12
## 1  ( 1 )  " "     " "    " "    " "     " "    " "     " "     " "
## 2  ( 1 )  " "     " "    " "    " "     " "    " "     " "     " "
## 3  ( 1 )  " "     " "    " "    " "     " "    " "     " "     " "
## 4  ( 1 )  " "     " "    " "    " "     " "    " "     " "     " "
## 5  ( 1 )  " "     " "    " "    " "     " "    " "     " "     " "
## 6  ( 1 )  " "     " "    " "    " "     " "    " "     " "     " "
## 7  ( 1 )  " "     " "    " "    " "     " "    " "     " "     " "
## 8  ( 1 )  " "     " "    " "    " "     " "    " "     " "     " "
## 9  ( 1 )  " "     " "    " "    " "     " "    " "     " "     " "
## 10  ( 1 ) " "     " "    " "    " "     " "    " "     " "     " "
## 11  ( 1 ) " "     " "    " "    " "     " "    " "     " "     " "
## 12  ( 1 ) " "     " "    " "    " "     " "    " "     " "     " "
## 13  ( 1 ) " "     " "    " "    " "     " "    " "     " "     " "
## 14  ( 1 ) " "     " "    " "    " "     " "    " "     " "     " "
## 15  ( 1 ) " "     " "    " "    " "     "*"    " "     " "     " "
## 16  ( 1 ) " "     " "    "*"    " "     "*"    " "     " "     " "
## 17  ( 1 ) " "     " "    "*"    " "     "*"    " "     "*"     " "
## 18  ( 1 ) " "     " "    "*"    " "     "*"    " "     "*"     " "
## 19  ( 1 ) " "     " "    "*"    " "     "*"    " "     "*"     " "
## 20  ( 1 ) " "     " "    "*"    " "     "*"    " "     "*"     " "
##           RAMNTALL NGIFTALL MAXRAMNT LASTGIFT TDON RFA_96_RF RFA_96_RL
## 1  ( 1 )  " "      " "      " "      "*"      " "  " "       " "
## 2  ( 1 )  " "      " "      " "      "*"      " "  " "       " "
## 3  ( 1 )  " "      " "      " "      "*"      " "  " "       " "
## 4  ( 1 )  " "      " "      "*"      "*"      " "  " "       " "
## 5  ( 1 )  " "      " "      "*"      "*"      " "  " "       " "
## 6  ( 1 )  " "      "*"      "*"      "*"      " "  " "       " "
## 7  ( 1 )  "*"      "*"      "*"      "*"      " "  " "       " "
## 8  ( 1 )  "*"      "*"      "*"      "*"      " "  " "       " "
## 9  ( 1 )  "*"      "*"      "*"      "*"      " "  " "       " "
## 10  ( 1 ) "*"      "*"      "*"      "*"      " "  " "       " "
## 11  ( 1 ) "*"      "*"      "*"      "*"      " "  " "       " "
```

```
## 12  ( 1 ) "*"      "*"      "*"      "*"      " "  " "       " "
## 13  ( 1 ) "*"      "*"      "*"      "*"      "*"  " "       " "
## 14  ( 1 ) "*"      "*"      "*"      "*"      "*"  " "       " "
## 15  ( 1 ) "*"      "*"      "*"      "*"      "*"  " "       " "
## 16  ( 1 ) "*"      "*"      "*"      "*"      "*"  " "       " "
## 17  ( 1 ) "*"      "*"      "*"      "*"      "*"  " "       " "
## 18  ( 1 ) "*"      "*"      "*"      "*"      "*"  " "       "*"
## 19  ( 1 ) "*"      "*"      "*"      "*"      "*"  " "       "*"
## 20  ( 1 ) "*"      "*"      "*"      "*"      "*"  " "       "*"
##           RFA_96_RN RFA_96_RS RFA_96_F2 RFA_96_F3 RFA_96_F4 RFA_96_AC
## 1  ( 1 )  " "       " "       " "       " "       " "       " "
## 2  ( 1 )  " "       " "       " "       " "       " "       " "
## 3  ( 1 )  " "       " "       " "       " "       " "       " "
## 4  ( 1 )  " "       " "       " "       " "       " "       " "
## 5  ( 1 )  " "       " "       " "       " "       " "       " "
## 6  ( 1 )  " "       " "       " "       " "       " "       " "
## 7  ( 1 )  " "       " "       " "       " "       " "       " "
## 8  ( 1 )  " "       " "       " "       " "       " "       " "
## 9  ( 1 )  " "       " "       " "       "*"       " "       " "
## 10  ( 1 ) " "       " "       " "       "*"       "*"       " "
## 11  ( 1 ) " "       " "       "*"       "*"       "*"       " "
## 12  ( 1 ) " "       " "       "*"       "*"       "*"       " "
## 13  ( 1 ) " "       " "       "*"       "*"       "*"       " "
## 14  ( 1 ) " "       " "       "*"       "*"       "*"       " "
## 15  ( 1 ) " "       " "       "*"       "*"       "*"       " "
## 16  ( 1 ) " "       " "       "*"       "*"       "*"       " "
## 17  ( 1 ) " "       " "       "*"       "*"       "*"       " "
## 18  ( 1 ) " "       " "       "*"       "*"       "*"       " "
## 19  ( 1 ) " "       " "       "*"       "*"       "*"       " "
## 20  ( 1 ) " "       " "       "*"       "*"       "*"       " "
##           RFA_96_AD RFA_96_AE RFA_96_AF RFA_96_AG
## 1  ( 1 )  " "       " "       " "       " "
## 2  ( 1 )  " "       " "       " "       "*"
## 3  ( 1 )  " "       " "       "*"       "*"
## 4  ( 1 )  " "       " "       "*"       "*"
## 5  ( 1 )  " "       "*"       "*"       "*"
## 6  ( 1 )  " "       "*"       "*"       "*"
## 7  ( 1 )  " "       "*"       "*"       "*"
## 8  ( 1 )  " "       "*"       "*"       "*"
## 9  ( 1 )  " "       "*"       "*"       "*"
## 10  ( 1 ) " "       "*"       "*"       "*"
## 11  ( 1 ) " "       "*"       "*"       "*"
## 12  ( 1 ) " "       "*"       "*"       "*"
## 13  ( 1 ) " "       "*"       "*"       "*"
## 14  ( 1 ) " "       "*"       "*"       "*"
## 15  ( 1 ) " "       "*"       "*"       "*"
## 16  ( 1 ) " "       "*"       "*"       "*"
## 17  ( 1 ) " "       "*"       "*"       "*"
## 18  ( 1 ) " "       "*"       "*"       "*"
## 19  ( 1 ) " "       "*"       "*"       "*"
## 20  ( 1 ) " "       "*"       "*"       "*"
```

Let see the best 4 variables model

```
coef(regfit.fwd,4)
```

```
## (Intercept)    MAXRAMNT    LASTGIFT   RFA_96_AF   RFA_96_AG
##  3.29362864  0.09793824  0.54277119  3.20658194  7.44388741
```

Fitting the model with 4 variables. The final model.

```
modelB2 = lm(DAMT ~ MAXRAMNT+LASTGIFT+RFA_96_F+RFA_96_A,data=regrData2,subset=trainIdx)
```

```
summary(modelB2)
```

```
##
## Call:
## lm(formula = DAMT ~ MAXRAMNT + LASTGIFT + RFA_96_F + RFA_96_A,
##     data = regrData2, subset = trainIdx)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -66.471  -1.979  -0.398   1.829 157.975
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.55263    8.24346   0.188  0.85062
## MAXRAMNT     0.10217    0.02245   4.550  5.6e-06 ***
## LASTGIFT     0.52130    0.02807  18.570  < 2e-16 ***
## RFA_96_F2   -0.94195    0.42345  -2.224  0.02620 *
## RFA_96_F3   -1.60658    0.49390  -3.253  0.00116 **
## RFA_96_F4   -1.67000    0.54258  -3.078  0.00211 **
## RFA_96_AC   -0.29999   10.07693  -0.030  0.97625
## RFA_96_AD    2.09855    8.23791   0.255  0.79894
## RFA_96_AE    3.68185    8.23908   0.447  0.65500
## RFA_96_AF    5.71148    8.24388   0.693  0.48849
## RFA_96_AG   10.24097    8.26230   1.239  0.21527
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.225 on 2716 degrees of freedom
## Multiple R-squared:  0.5732, Adjusted R-squared:  0.5717
## F-statistic: 364.8 on 10 and 2716 DF,  p-value: < 2.2e-16
```

```
coef(modelB2)
```

```
## (Intercept)    MAXRAMNT    LASTGIFT   RFA_96_F2   RFA_96_F3   RFA_96_F4
##   1.5526340   0.1021712   0.5213016  -0.9419531  -1.6065777  -1.6699977
##   RFA_96_AC   RFA_96_AD   RFA_96_AE   RFA_96_AF   RFA_96_AG
##  -0.2999920   2.0985460   3.6818535   5.7114762  10.2409702
```

Checking collinearility. Less than 10 so it seems ok.

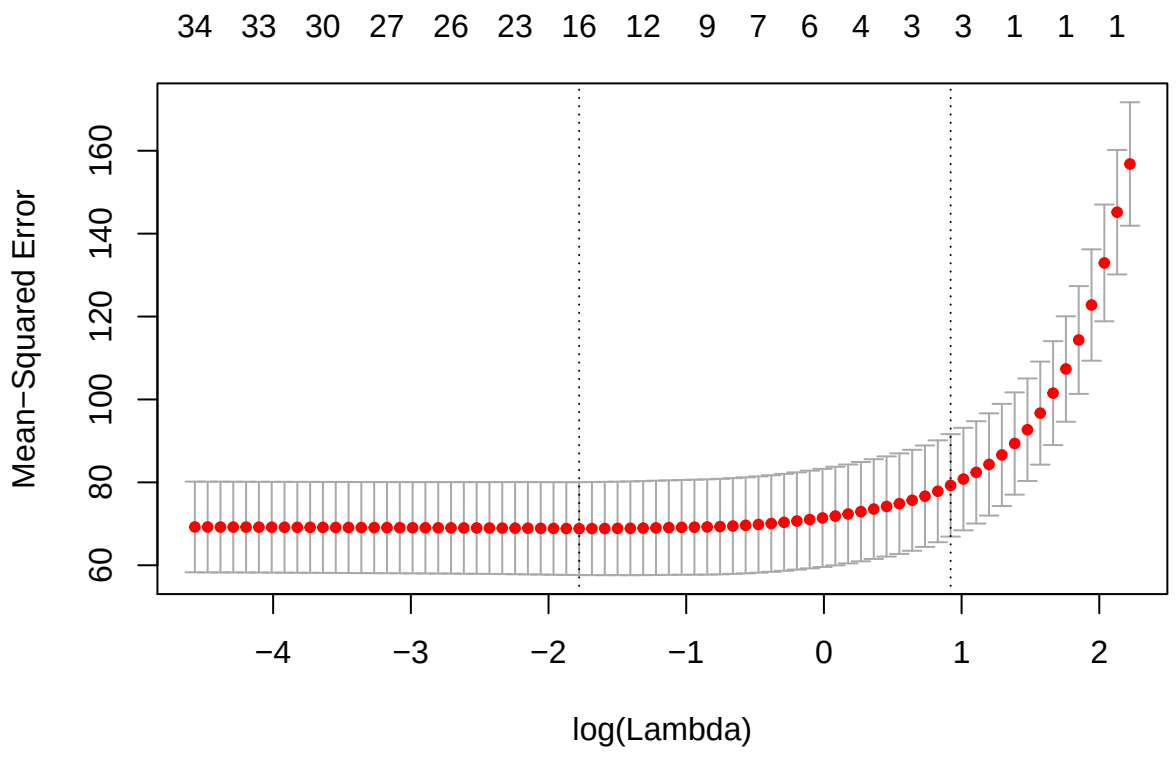
```
vif(modelB2)
```

```
##              GVIF Df GVIF^(1/(2*Df))
## MAXRAMNT 3.982433  1        1.995603
## LASTGIFT 4.427870  1        2.104250
## RFA_96_F 1.690494  3        1.091446
## RFA_96_A 2.701261  5        1.104477
```

The main function in this package is glmnet(), which can be used glmnet() to fit ridge regression models, lasso models, and more. In particular, we must pass in an x matrix as well as a y vector.

c. Shrinkage models (ISLR Section 6.2) or Principal Components Regressions (ISLR Section 6.3)

```
regX = model.matrix(DAMT ~ .-ID,data=regrData2)[,-1]
regY = regrData2$DAMT
cvLasso = cv.glmnet(regX[trainIdx,],regY[trainIdx],alpha=1)
plot(cvLasso)
```




```
modelC1 = glmnet(regX[trainIdx,],regY[trainIdx],alpha=1,lambda=cvLasso$lambda.min)
coef(modelC1)
```

```
## 36 x 1 sparse Matrix of class "dgCMatrix"
##                        s0
```

```
## (Intercept)  7.035515546
## AGE           .
## HOME1        -0.552665884
## HINC2        -0.099016689
## HINC3         .
## HINC4         .
## HINC5         0.018556182
## HINC6         .
## HINC7         .
## HINC0         .
## GENDERM       .
## GENDERU       .
## MEDAGE        .
## MEDPPH        .
## MEDHVAL       .
## MEDINC        .
## MEDEDUC       .
## NUMPROM       .
## NUMPRM12      .
## RAMNTALL      0.005115799
## NGIFTALL     -0.093753661
## MAXRAMNT      0.080689384
## LASTGIFT      0.531888146
## TDON          0.031993443
## RFA_96_RF     0.157845384
## RFA_96_RL     0.109024950
## RFA_96_RN     .
## RFA_96_RS    -0.330455821
## RFA_96_F2     .
## RFA_96_F3    -0.797464142
## RFA_96_F4    -0.716032186
## RFA_96_AC     .
## RFA_96_AD    -2.363480771
## RFA_96_AE    -1.314834040
## RFA_96_AF     .
## RFA_96_AG     4.101114776
```

In lasso the resulting coefficient estimates are sparse. So the resulting lasso model contains only six variables

```
bestlam=cvLasso$lambda.min
lasso.coef=predict (modelC1 ,type ="coefficients",s=bestlam )[1:20 ,]
lasso.coef
```

```
##  (Intercept)          AGE        HOME1        HINC2        HINC3
##  7.035515546  0.000000000 -0.552665884 -0.099016689  0.000000000
##        HINC4        HINC5        HINC6        HINC7        HINC0
##  0.000000000  0.018556182  0.000000000  0.000000000  0.000000000
##     GENDERM      GENDERU       MEDAGE       MEDPPH      MEDHVAL
##  0.000000000  0.000000000  0.000000000  0.000000000  0.000000000
##       MEDINC      MEDEDUC      NUMPROM     NUMPRM12     RAMNTALL
##  0.000000000  0.000000000  0.000000000  0.000000000  0.005115799
```

```
lasso.coef[lasso.coef !=0]
```

```
##  (Intercept)        HOME1        HINC2        HINC5     RAMNTALL
##  7.035515546 -0.552665884 -0.099016689  0.018556182  0.005115799
```

```
modelC2 = glmnet(regX[trainIdx,],regY[trainIdx],alpha=1,lambda=cvLasso$lambda.1se)
coef(modelC2)
```

```
## 36 x 1 sparse Matrix of class "dgCMatrix"
##                      s0
## (Intercept) 7.09133651
## AGE         .
## HOME1       .
## HINC2       .
## HINC3       .
## HINC4       .
## HINC5       .
## HINC6       .
## HINC7       .
## HINC0       .
## GENDERM     .
## GENDERU     .
## MEDAGE      .
## MEDPPH      .
## MEDHVAL     .
## MEDINC      .
## MEDEDUC     .
## NUMPROM     .
## NUMPRM12    .
## RAMNTALL    .
## NGIFTALL    .
## MAXRAMNT    0.02546891
## LASTGIFT    0.53543985
## TDON        .
## RFA_96_RF   .
## RFA_96_RL   .
## RFA_96_RN   .
## RFA_96_RS   .
## RFA_96_F2   .
## RFA_96_F3   .
## RFA_96_F4   .
## RFA_96_AC   .
## RFA_96_AD   .
## RFA_96_AE   .
## RFA_96_AF   .
## RFA_96_AG   0.46866380
```

In this model only intercept is used?

```
bestlam=cvLasso$lambda.1se
lasso.coef=predict (modelC2 ,type ="coefficients",s=bestlam )[1:20 ,]
lasso.coef
```

```
## (Intercept)         AGE       HOME1       HINC2       HINC3       HINC4
##    7.091337    0.000000    0.000000    0.000000    0.000000    0.000000
##       HINC5       HINC6       HINC7       HINC0     GENDERM     GENDERU
##    0.000000    0.000000    0.000000    0.000000    0.000000    0.000000
##      MEDAGE      MEDPPH     MEDHVAL      MEDINC     MEDEDUC     NUMPROM
##    0.000000    0.000000    0.000000    0.000000    0.000000    0.000000
##    NUMPRM12    RAMNTALL
##    0.000000    0.000000
```

```
lasso.coef[lasso.coef !=0]
```

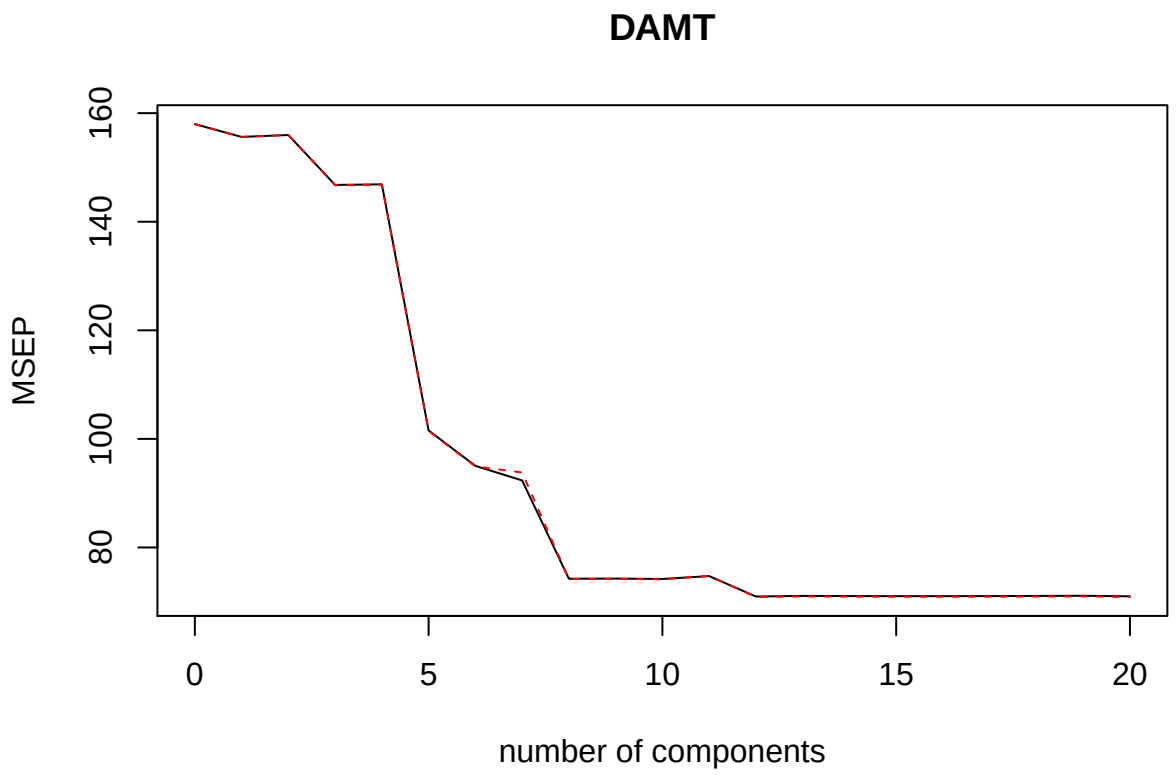
```
## (Intercept)
##    7.091337
```

d. Another model of your choice, which may include a second model from one of the three prior categories I will illustrate Principal Components Regression here.

```
pcrFit=pcr(DAMT~.-ID,data=regrData2,subset=trainIdx,ncomp=20,validation ="CV")
summary(pcrFit)
```

```
## Data:    X dimension: 2727 35
##  Y dimension: 2727 1
## Fit method: svdpc
## Number of components considered: 20
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV           12.57    12.47    12.49    12.12    12.12    10.08    9.748
## adjCV        12.57    12.47    12.49    12.11    12.12    10.07    9.743
##        7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
## CV       9.610    8.616    8.618     8.614     8.644     8.424     8.431
## adjCV    9.686    8.613    8.615     8.610     8.646     8.419     8.425
##        14 comps  15 comps  16 comps  17 comps  18 comps  19 comps
## CV        8.429     8.428     8.428     8.429     8.430     8.432
## adjCV     8.423     8.422     8.422     8.422     8.424     8.426
##        20 comps
## CV        8.426
## adjCV     8.419
##
## TRAINING: % variance explained
##       1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps
## X      97.177   98.524   99.733   99.901    99.94    99.96    99.97
## DAMT    1.523    1.525    7.761    7.789    37.23    41.84    43.40
##       8 comps  9 comps  10 comps  11 comps  12 comps  13 comps  14 comps
## X       99.99    99.99    100.00    100.00    100.00    100.00    100.00
## DAMT    53.70    53.70     53.85     53.85     56.11     56.12     56.17
##       15 comps  16 comps  17 comps  18 comps  19 comps  20 comps
## X       100.00    100.00    100.00    100.00    100.00    100.00
## DAMT     56.22     56.25     56.25     56.31     56.33     56.43
```

```
validationplot(pcrFit,val.type="MSEP")
```



The variance explained at 8 components is 53.70% and at 12 components is 56.11%.

```
modelD1 = pcr(DAMT~.-ID,data=regrData2,subset=trainIdx,ncomp=8)
summary(modelD1)
```

```
## Data:    X dimension: 2727 35
##  Y dimension: 2727 1
## Fit method: svdpc
## Number of components considered: 8
## TRAINING: % variance explained
##       1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps
## X      97.177   98.524   99.733   99.901    99.94    99.96    99.97
## DAMT    1.523    1.525    7.761    7.789    37.23    41.84    43.40
##       8 comps
## X       99.99
## DAMT    53.70
```

5. Model Validation

```
calcMSE = function(model,modelLabel,dataSet,trainIdx,newX=NULL,ncomp=NULL)
{
  # The predict method for glmnet will need to be called differently from the
```

```
  # other predict methods.
  if ("glmnet" %in% class(model)) {
    predVals = predict(model,newX,type="response")
  } else if ("mvr" %in% class(model)) {
    predVals = predict(model,dataSet,ncomp=ncomp)
  } else {
    predVals = predict(model,dataSet)
  }
  MSE = list(
    name = modelLabel,
    train = mean( (predVals[trainIdx] - dataSet$DAMT[trainIdx])^2 ),
    test = mean( (predVals[-trainIdx] - dataSet$DAMT[-trainIdx])^2 )
  )

  return(MSE)
}

numModels = 6   # number of models that I have fit (A1, B1, B2, C1, C2, and D1)
modelMSEs = data.frame(
  Model = rep(NA,numModels),
  Train.MSE = rep(NA,numModels),
  Test.MSE = rep(NA,numModels)
  )

modelMSEs[1,] = calcMSE(modelA1,"A1",regrData2,trainIdx)
modelMSEs[2,] = calcMSE(modelB1,"B1",regrData2,trainIdx)
modelMSEs[3,] = calcMSE(modelB2,"B2",regrData2,trainIdx)
modelMSEs[4,] = calcMSE(modelC1,"C1",regrData2,trainIdx,newX=regX)
modelMSEs[5,] = calcMSE(modelC2,"C2",regrData2,trainIdx,newX=regX)
modelMSEs[6,] = calcMSE(modelD1,"D1",regrData2,trainIdx,ncomp=8)

print(modelMSEs)
```

```
##   Model Train.MSE  Test.MSE
## 1    A1  72.62404  58.98824
## 2    B1  66.21256  60.44167
## 3    B2  67.37275  65.91447
## 4    C1  66.84264  62.31209
## 5    C2  78.26020  61.12234
## 6    D1  73.08420 201.56699
```

I think the model B1 is the best model with the lowest bias as well as variance of the errors. Althoud both models C1 and C2 get pretty close. C2 contains less variables than any of the other models but B1 has lower Test MSE.