STAT 652: Part 2

Arin Ghosh

12/4/2018

Part 2: Predicting HUIDHSI (Regression Analysis)

1 Data

1.1 Data Loading

Again the first step is to load the appropriate dataset into the R Studio environment. The dataset can be found on the project github repository. Once downloaded in to the working directory of the R Studio, we can load the data using read.csv() command. Since the data file is big, I have used the `data.table` library to load the data quickly. It is generally faster than the `read.csv()` that is found in the base model. One difference between the default read.csv() and `data.table::fread()` is that we need to explicitly pass `stringAsFactor` in order to make sure that the data is read properly.

```
library(data.table, quietly = T)
hs <- fread("HStrain.csv", stringsAsFactors = T)
dim(hs)
```

```
## [1] 10000   591
```

```
# how many different types of measurements are represented by all these columns?
cn <- colnames(hs)
table(substr(cn,start=1,stop=3))
```

```
##
## ADL ADM ALC CAG CCC CGE CIH CR1 CR2 DHH DPS DS2 EDU FAL GEN GEO HC2 HUI
##   4   4   5  45  31   8  27  34  19   9  33   4   2  15  10   2  19   1
## HUP HWT IAL IN2 LBF LON MED NUR OH3 OWN PA2 RET RPL SDC SLP SLS SMK SPA
##   1   5   6   8  19   4  33  12  27   2  49  32  19   6   1   6  21  24
## SSA TRA
##  25  19
```

We can quickly see that there are 10000 rows while the number of columns are 591. We can also see that a total of 38 diffrent measurements that are represented in the dataset and each group (except 3) further subdivided into subgroups. Therefore we spend a considerable amount of time trying to figure out how can we reduce this number to something that is managable while keeping the overall variance of the data set intact. Upon a careful look we can see that there are some variables that are used for record keeping (ID type) and kept it in the data set for administrative purposes (e.g. group of variables starts with "ADM"), we need to first remove it.

In class we've taught about the 'curse of dimensionality' which states that the more dimensions you work with, the less effective standard computational and statistical techniques become. This has repercussions that need some serious workarounds when machines are dealing with Big Data. In general, I have used the following techniques to reduce the number of predictors from what was given to us to start with

- Step 1: I have gone through the details of the data documentation that was given to us. The document shows that some group of variables are summarized as a single variable. I have chosen those variables

since they represent a good balance of the group of variables without loosing a lot of variability. For example, ADLDCLS is an overall summary measurement of Instrumental and Basic Activities of Daily Living for a respondent. Also not all of the data groups that we want to include, has a summary variable. So in that case we have derived Multiple Correspondence Analysis (MCA) (since they are categorical variables) and included the dimensions that are explaining the variables significantly. Please refer appendix for details

- Step 2: Once we have finished filtering the columns manually by going through the dataset and through MCA technique, the next thing we want to do is to run a subset selection technique. We used two subset selection techniques
  - Regression Subset Selection Technique - Forward, Backward, Mixed
  - Lasso Subset Selection - Used different lambda values and then selected the best lambda for which the mean error is the lowest.
  - Finally compared both of these techniques, and since Lasso gave me the best result, and went with the columns that was ultimately filtered out.

After going through these process of predictor reduction, we finally choose 31 columns out of the possible 590 columns.

```
## [1] "ADLDCLSNO FUNC IMPAIR"       "ADLDCLSSEV IMPAIRMENT"
## [3] "ADLDCLSTOTAL IMPAIRMENT"     "CCCF1HAS NO CHRON CON"
## [5] "CR1FRHCREC FORMAL H C"       "CR2DTHCDID NOT REC H C"
## [7] "CR2DFARREG BASIS DLY"        "EDUDR04POST-SEC. GRAD."
## [9] "FALG022"                             "FALG027"
## [11] "FALG02NOT APPLICABLE"       "GENDHDIFAIR"
## [13] "GENDHDIPOOR"                    "GENDMHIFAIR"
## [15] "GENDMHIGOOD"                    "GENDMHIPOOR"
## [17] "HUPDPADPAIN ATT. LEV.2"     "HUPDPADPAIN ATT. LEV.3"
## [19] "HUPDPADPAIN ATT. LEV.4"     "HUPDPADPAIN ATT. LEV.5"
## [21] "IN2GHH$80,000 OR MORE"      "IN2GHH< $20,000"
## [23] "LONDSCR"                          "NURDHNRNOT AT HIGH N R"
## [25] "PA2DSCR"                          "SLSDCLSEXT DISSATISFIED"
## [27] "SLSDCLSEXT SATISFIED"       "SLSDCLSSATISFIED"
## [29] "SLSDCLSSL DISSATISFIED"     "SPAFPARPARTICIPANT"
## [31] "CIH.Dim.4"
```

Methods

Before we could run any model, we first need to split the dataset into train/test. For that I have used 70:30 split. Based on the filtered columns, I have build three regression models. They are: * Linear Regression * Random Forest * Gradient Boosting Model (GBM)

In terms of the error rate, the linear regression model gave me the lowest root mean square error, therefore that is what I chose as my final model in terms of predicting `HUIDHSI`. Here is the summary of the final model.

```
Call:
lm(formula = HUIDHSI ~ ., data = hsred2.train)

Residuals:
     Min       1Q   Median       3Q      Max
-0.94931 -0.03026  0.02223  0.05567  0.45694

Coefficients:
```

```
                          Estimate Std. Error t value
(Intercept)               0.831882   0.001359 612.244
ADLDCLSMOD.IMPAIRMENT    -0.016032   0.001520 -10.545
ADLDCLSNO.FUNC.IMPAIR     0.028790   0.001769  16.271
ADLDCLSSEV.IMPAIRMENT    -0.017453   0.001468 -11.890
ADLDCLSTOTAL.IMPAIRMENT  -0.007804   0.001433  -5.446
CCCF1HAS.NO.CHRON.CON     0.005390   0.001410   3.822
CR1FRHCREC.FORMAL.H.C    -0.004662   0.001756  -2.655
CR2DTHCDID.NOT.REC.H.C    0.009019   0.001974   4.568
CR2DFARREG.BASIS.DLY     -0.004080   0.001539  -2.650
EDUDR04POST.SEC..GRAD.    0.004840   0.001419   3.410
FALG022                  -0.003912   0.001501  -2.607
FALG027                  -0.006167   0.001383  -4.458
FALG02NOT.APPLICABLE      0.006580   0.001572   4.186
GENDHDIFAIR              -0.008038   0.001516  -5.302
GENDHDIPOOR              -0.009822   0.001552  -6.329
GENDMHIFAIR              -0.017192   0.001487 -11.563
GENDMHIGOOD              -0.009509   0.001431  -6.646
GENDMHIPOOR              -0.011840   0.001393  -8.498
HUPDPADPAIN.ATT..LEV.2   -0.013663   0.001375  -9.935
HUPDPADPAIN.ATT..LEV.3   -0.037400   0.001400 -26.722
HUPDPADPAIN.ATT..LEV.4   -0.072251   0.001428 -50.605
HUPDPADPAIN.ATT..LEV.5   -0.111778   0.001537 -72.711
IN2GHH.80.000.OR.MORE     0.001835   0.001451   1.264
LONDSCR                  -0.010775   0.001533  -7.029
NURDHNRNOT.AT.HIGH.N.R    0.003714   0.001434   2.590
PA2DSCR                   0.007354   0.001534   4.793
SLSDCLSEXT.DISSATISFIED  -0.004990   0.001462  -3.414
SLSDCLSEXT.SATISFIED      0.014593   0.001923   7.587
SLSDCLSSATISFIED          0.011573   0.001942   5.959
SLSDCLSSL.DISSATISFIED   -0.003132   0.001512  -2.071
SPAFPARPARTICIPANT        0.004148   0.001389   2.986
CIH.Dim.4                -0.004077   0.001379  -2.958
                         Pr(>|t|)
(Intercept)               < 2e-16 ***
ADLDCLSMOD.IMPAIRMENT     < 2e-16 ***
ADLDCLSNO.FUNC.IMPAIR     < 2e-16 ***
ADLDCLSSEV.IMPAIRMENT     < 2e-16 ***
ADLDCLSTOTAL.IMPAIRMENT 5.33e-08 ***
CCCF1HAS.NO.CHRON.CON    0.000133 ***
CR1FRHCREC.FORMAL.H.C    0.007940 **
CR2DTHCDID.NOT.REC.H.C   5.00e-06 ***
CR2DFARREG.BASIS.DLY     0.008057 **
EDUDR04POST.SEC..GRAD.   0.000652 ***
FALG022                  0.009163 **
FALG027                  8.39e-06 ***
FALG02NOT.APPLICABLE     2.88e-05 ***
GENDHDIFAIR              1.18e-07 ***
GENDHDIPOOR              2.61e-10 ***
GENDMHIFAIR               < 2e-16 ***
GENDMHIGOOD              3.23e-11 ***
GENDMHIPOOR               < 2e-16 ***
HUPDPADPAIN.ATT..LEV.2    < 2e-16 ***
HUPDPADPAIN.ATT..LEV.3    < 2e-16 ***
```

```
HUPDPADPAIN.ATT..LEV.4   < 2e-16 ***
HUPDPADPAIN.ATT..LEV.5   < 2e-16 ***
IN2GHH.80.000.OR.MORE   0.206098
LONDSCR                 2.27e-12 ***
NURDHNRNOT.AT.HIGH.N.R  0.009624 **
PA2DSCR                 1.68e-06 ***
SLSDCLSEXT.DISSATISFIED 0.000644 ***
SLSDCLSEXT.SATISFIED    3.70e-14 ***
SLSDCLSSATISFIED        2.65e-09 ***
SLSDCLSSL.DISSATISFIED  0.038407 *
SPAFPARPARTICIPANT      0.002838 **
CIH.Dim.4               0.003110 **
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.1136 on 6968 degrees of freedom
Multiple R-squared:  0.7241,    Adjusted R-squared:  0.7228
F-statistic: 589.8 on 31 and 6968 DF,  p-value: < 2.2e-16
```

For the final linear regression model that I chose, all of the 31 predictors are significant except the Income column `(IN2GHH.80.000.OR.MORE)`. The calculated RMSE is 0.01259476 with Multiple R-Square value of 0.72 means this model explains 72% of the available data and a p-value of p-value: $<$ 2.2e-16 which means its a significant model in predicting `PA2DSCR`. Lets take an example of how we can interprete this model and the relationship between the dependent and the independent variable. For example we see the coefficient for `LONDSCR = -0.010775` in the model which means that for one unit increase in the Lonliness scale, keeping all other predictors constant, it contributes to -0.010775 times decrease in HUIDHSI score which means the more lonely the person is, the more unhealthy the person will become. So we can now not only measure qualitative relations but also quantitatively measure the relationship between the dependent and independent variables. We can also compare different predictors and say which predictor is more impacting in determining the `HUIDHSI` score.

Results

In order to evaluate different models that I build, I used the measurement of root mean square error for the purpose since this is a regression problem, where we are predicting a real number rather than a categorical value. The root mean square derivation is frequently used in the world of statistics. Mean Square Error (MSE) is the variance of the residuals (prediction errors). Residuals are a measure of how far from the regression line data points are; RMSE is a measure of how spread out these residuals are. In other words, it tells you how concentrated the data is around the line of best fit. The lower the number, the better the model is. The minimum the RMSE can go is 0. Therefore a value very close to zero is considered a good model.

Here are the three MSE that I have got for different models that are trained with the train set and evaluated on the test set:

Table 1: Mean Square Error For Different Models

Linear Reg	Random Forest	Gradient Boosting Method
0.0125948	0.0141791	0.0384641

Based on the result, we can see that our best model that was able to predict the test set with the least error

is *Linear Regression* followed by the model built using Random Forest. We therefore conclude that Linear Regression is the best model along with the selected set of predictors to predict `HUIDHSI` score.

Evaluation of best model on the Holdout dataset

Once I finalized the model, I have re-fit the model again on the full dataset (without any tran-test split) and then used the holdout dataset to measure the MSE which I got is **0.01533271**

Conclusion

After going through all the models, we have picked the linear regression model with four predictors as our final selected model. However we have only considered linear terms in this model, which can be a drawback in this model. We also haven't extensively used K-Fold cross validation here due to time complexity. Therefore in the future it will be noteworthy to check how adding interaction and non-linear terms and doing K-fold CV changes the predictive ability of all these models that I have tried, or if there is any other family of regression techniques stands out when these new terms / new data are introduced.

Apprendix (Part II)

Here is a list of all the libraries that are used for part II. Load all the required Libraries

```
req_libs = c("data.table", 'FactoMineR', 'factoextra', 'dplyr',
         'leaps', 'glmnet', 'randomForest', 'gbm')
lapply(req_libs, require, character.only = TRUE)
```

Load Data Set:

```
hs <- fread("HStrain.csv", stringsAsFactors = T)
```

Explore the dataset.

```
cn <- colnames(hs)
length(unique(substr(cn,start=1,stop=3)))
```

```
## [1] 38
```

```
table(substr(cn,start=1,stop=3))
```

```
##
## ADL ADM ALC CAG CCC CGE CIH CR1 CR2 DHH DPS DS2 EDU FAL GEN GEO HC2 HUI
##   4   4   5  45  31   8  27  34  19   9  33   4   2  15  10   2  19   1
## HUP HWT IAL IN2 LBF LON MED NUR OH3 OWN PA2 RET RPL SDC SLP SLS SMK SPA
##   1   5   6   8  19   4  33  12  27   2  49  32  19   6   1   6  21  24
## SSA TRA
##  25  19
```

Remove unwanted columns. * The variables that start with ADM are to do with administering the survey and are not useful for prediction. For example, ADM_RNO is a sequential record number, ADM_N09 indicates whether the interview was by phone, in-person, etc. These columns do not add any value to the model's predicting abilities, so we will remove these.

```
hsred <- dplyr::select(hs,-starts_with("ADM"))
```

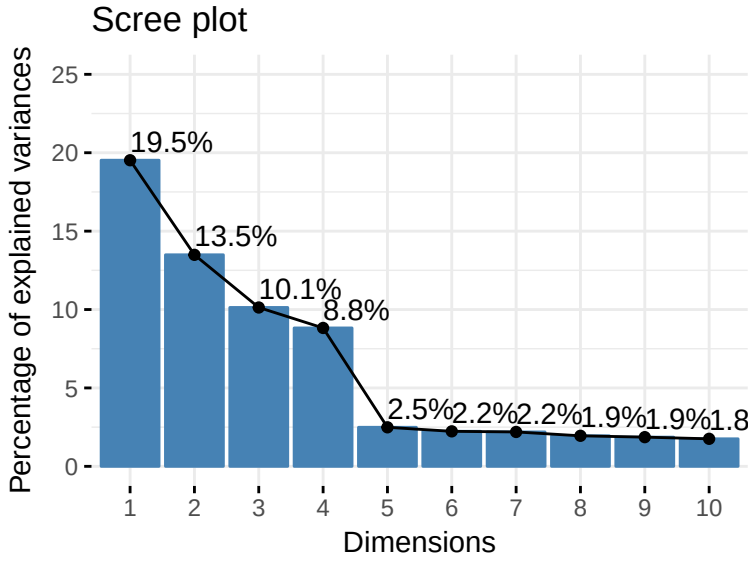
The next step for me was to go through the data documentation and search for predictors that makes sense. I need to admit here that I got a little help from the lecture notes on this. The instructor has provided a fairly good list of predictors which has an overall summary indicator, however I have included couple others that I felt was important as well.

```
hsred <- dplyr::select(hs,
ADLDCLS, # Instrumental and Basic Activities of Daily Living Classification
ALCDTTM, # Type of Drinker
CAGDFAP, #This variable indicates the frequency of assistance provided by the respondent to the main ca
CCCF1, # Has a Chronic Condition
CCCDCPD, #Has Chronic Obstructive Pulmonary Disease
CR1FRHC,# Flag for Receiving Formal Home Care Services
CR2DTHC, # Receipt of Formal or Informal Home Care
CR2DFAR, # Frequency of Assistance Received from the Main Caregiver (for the main source of assistance)
```

```
DPSDSF, # Depression Scale - Probability of Caseness to Respondents
EDUDR04, # Highest Level of Education - Household, 4 Levels
FALG02, # Number of falls - past 12 months - grouped
GENDHDI, # Perceived Health
GENDMHI, # Perceived Mental Health
HC2FCOP, # Flag for Consultation with Health Professional
HUPDPAD, # Health Utilities Index
HWTGBMI, # Body mass index - grouped
IN2GHH, # Total Household Income - All Sources - grouped
LONDSCR, # Three Item Loneliness Scale - Score
MEDF1, # Flag Indicating Medication Use (Past Month)
NURDHNR, # High Nutritional Risk
PA2DSCR, # PASE Score
SLSDCLS, # Satisfaction with Life Scale
SMKDSTY, # Number of Years Since Stopped Smoking Completely
SPAFPAR, # Frequency of Community-Related Activity Participation (participant)
GEOGCMA2, # Metropilitan Area Summary
HUIDHSI # response
)
```

There were some predictors which do not have any sort of summary variable, therefore for those variables I have done a quick MCA() analysis to come up with a custom columns. I have first written a small function, that lets me explore the result of the analysis and then I used the output of the function to select the number of dimensions that I want to include in the final table. Lets explore the group of variables starting with "CIH":

```
result.mca <- MCA(dplyr::select(hs,starts_with("CIH")), graph = F)
fviz_screeplot(result.mca, ylim = c(0,25), addlabels = TRUE)
```

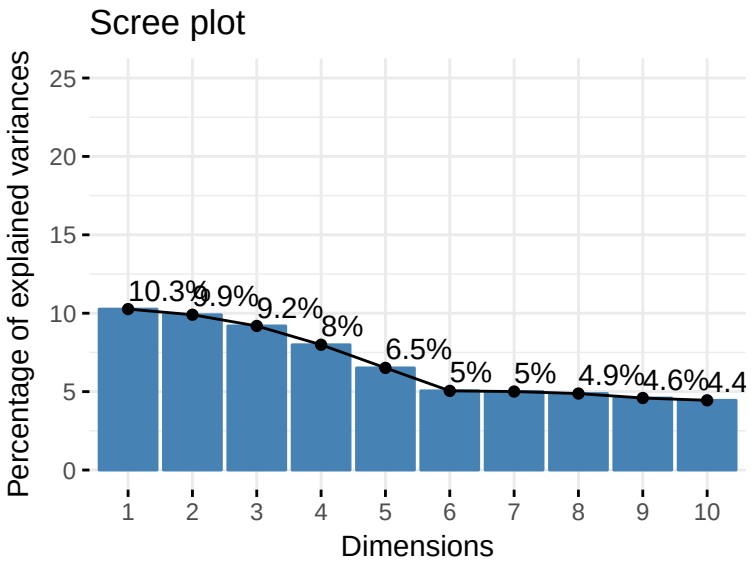



```
# looks like the first 4 Dims exaplain about 50% of the data
CIHPCs <- result.mca$ind$coord[,1:4]
colnames(CIHPCs) <- paste("CIH",colnames(CIHPCs))

# include these new predictors in the final dataset for model building
hsred <- data.frame(hsred,CIHPCs)
```

Also take a look into the variables starting with "DS2":

```
result.mca <- MCA(dplyr::select(hs,starts_with("DS2")), ncp = 10, graph = F)
fviz_screeplot(result.mca, ylim = c(0,25), addlabels = TRUE)
```




```
# looks like the first 7 Dims exaplain about 50% of the data
DS2PCs <- result.mca$ind$coord[,1:7]
colnames(DS2PCs) <- paste("DS2",colnames(DS2PCs))

# include these new predictors in the final dataset for model building
hsred <- data.frame(hsred,DS2PCs)
dim(hsred)
```

```
## [1] 10000    37
```

hsread now contains our final set of columns that we have chosen manually. There are 37 columns and 10000 rows. After manual selection, we can now run a subset selection and Lasso analysis to choose the most significant columns out of these 37 columns and proceed in building the model.

Regression Subset Selection

We first need to scale the dataset since some of the techniques that we are going to use requires that, split the data set into train test

```
tem <- model.matrix(HUIDHSI ~ .,data=hsred)[,-1]
X <- as.data.frame(scale(tem))
Y <- hsred$HUIDHSI
rm(tem)

# split test-train
set.seed(19)
n.train <- 7000
train <- sample(1:nrow(hs),replace=FALSE,size=n.train)
```

```
X.train <- X[train,]
Y.train <- Y[train]
X.test <- X[-train,]
Y.test <- Y[-train]
```

```
rr <- regsubsets(X.train,Y.train,nvmax=30, method="forward")
```

```
## Reordering variables and trying again:
```

```
ss <- summary(rr)
pbest <- which.min(ss$bic)
cols<- ss$which[pbest,-1] # don't include intercept
Xred <- as.matrix(X.test[,cols])
pred.test <- cbind(1,Xred) %*% coef(rr,id=26)
```

The `regsubsets()` choose 26 columns and came up with a MSE of **0.0414928**. Lets try and see if we can get a better result with Lasso technique that we learned in the class.

Lasso Subset Selection

```
lambdas <- 10^{seq(from=-3,to=5,length=100)} # range of lambdas to try from
alpha = 0.8
cv.lafit <- cv.glmnet(as.matrix(X.train),Y.train,alpha= alpha,lambda=lambdas) # one hot encoding
la.best.lam <- cv.lafit$lambda.1se
ll <- glmnet(as.matrix(X.train),Y.train,alpha= alpha,lambda=la.best.lam)
pred.test <- predict(ll,as.matrix(X.test))
```

Since the MSE we found here 0.0128092 is less than that of the regression subset, we chose the result produced by lasso techniques as our final set of columns. Which are:

\begin{center} **Final Set of Selected Predictors** \end{center}

```
##  [1] "ADLDCLSNO FUNC IMPAIR"   "ADLDCLSSEV IMPAIRMENT"
##  [3] "ADLDCLSTOTAL IMPAIRMENT" "CCCF1HAS NO CHRON CON"
##  [5] "CR1FRHCREC FORMAL H C"   "CR2DTHCDID NOT REC H C"
##  [7] "CR2DFARREG BASIS DLY"    "EDUDR04POST-SEC. GRAD."
##  [9] "FALG022"                 "FALG027"
## [11] "FALG02NOT APPLICABLE"    "GENDHDIFAIR"
## [13] "GENDHDIPOOR"             "GENDMHIFAIR"
## [15] "GENDMHIGOOD"             "GENDMHIPOOR"
## [17] "HUPDPADPAIN ATT. LEV.2"  "HUPDPADPAIN ATT. LEV.3"
## [19] "HUPDPADPAIN ATT. LEV.4"  "HUPDPADPAIN ATT. LEV.5"
## [21] "IN2GHH$80,000 OR MORE"   "IN2GHH< $20,000"
## [23] "LONDSCR"                 "NURDHNRNOT AT HIGH N R"
## [25] "PA2DSCR"                 "SLSDCLSEXT DISSATISFIED"
## [27] "SLSDCLSEXT SATISFIED"    "SLSDCLSSATISFIED"
## [29] "SLSDCLSSL DISSATISFIED"  "SPAFPARPARTICIPANT"
## [31] "CIH.Dim.4"
```

So we can now finally create our train and test data based on these 32 filtered columns.

```
nonz <- (as.numeric(coef(ll))!=0)[-1] # rm intercept
train.data <- data.frame(HUIDHSI=Y.train,X.train[,nonz])
test.data <- data.frame(HUIDHSI=Y.test,X.test[,nonz])

# remove all the unwanted temporary variables at this stage
rm(hsred, hs)
```

Models:

Linear Regression

```
linear.fit <- lm(HUIDHSI ~ ., data = train.data)
preds <- predict(linear.fit, newdata = test.data)
with(test.data, mean((test.data$HUIDHSI-preds)^2))
```

```
## [1] 0.01258929
```

Random Forest

```
set.seed(19)
bb <- randomForest(X.train,y=Y.train,xtest=X.test,
          ytest=Y.test,ntree=200,
mtry=sqrt(ncol(X.train)),importance=TRUE)
pred.test <- bb$test$predicted
mean((Y.test - pred.test)^2)
```

GBM

```
hs.train <- data.frame(HUIDHSI=Y.train,X.train)
hboost <- gbm(HUIDHSI ~ ., data=hs.train ,n.trees=400,
              interaction.depth=5, distribution="gaussian")
hs.test <- data.frame(HUIDHSI=Y.test,X.test)
pred.test <- predict(hboost,newdata=hs.test, n.trees=200,type="response")
mean((Y.test-pred.test)^2)
```

From the MSE score we can see that our best model is Linear Regression.