Homework 3

Wenjie Du

Please read all instructions carefully. Try to knit the file before final submission to catch any bugs. To allow this file to be knit, you might need to save it in a new location, as you do not have write permissions in the dropbox folder, and R does not work well with Chinese characters.
Some important steps: 1. Rename this file to your blackboard username. This lets us grade your assignment. 2. This assignment requires you to submit an RData file generated by this script. See the assignment sheet and video for details on how to do this. Submissions that do not include the correct files will be penalized. 3. Please do not create new columns in the data frame - you can include things like polynomials directly in the model formula.

Question 1A:

```
#TRAIN AND ESTIMATE YOUR FIRST MODEL (WITHOUT REPORTS) HERE
local({r <- getOption("repos")
       r["CRAN"] <- "http://cran.r-project.org"
       options(repos=r)})

#Last line should look something like
#model1A = lm(card~expenditure+owner,data=fullFile)
# setwd('D:/Dropbox/Teaching Lectures/Assignments/Homework 3')
stuData = read.csv('C:\\Users\\92998\\OneDrive\\Desktop\\Simon\\Predictive\\Student_Data_3.csv')
str(stuData) # have a look at data
```

```
## 'data.frame':    19785 obs. of  12 variables:
##  $ card       : logi  TRUE TRUE FALSE TRUE TRUE TRUE ...
##  $ reports    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ age        : num  37.7 33.2 33.7 30.5 32.2 ...
##  $ income     : num  4.52 2.42 4.5 2.54 9.79 ...
##  $ share      : num  0.03327 0.00522 0.00416 0.06521 0.06705 ...
##  $ expenditure: num  124.98 9.85 15 137.87 546.5 ...
##  $ owner      : int  1 0 1 0 1 0 0 1 1 0 ...
##  $ selfemp    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ dependents : int  3 3 4 0 2 0 2 0 0 0 ...
##  $ months     : int  54 34 58 25 64 54 7 77 97 65 ...
##  $ majorcards : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ active     : int  12 13 5 7 5 1 5 3 6 18 ...
```

```
# without reports

stuData$reports = NULL
View(stuData)
isTraining <- runif(nrow(stuData)) < .8
stuDataTrain <- subset(stuData, isTraining)
stuDataValid <- subset(stuData, !isTraining)
basicLM <- lm(card~., data = stuDataTrain)
summary(basicLM)
```

```
##
## Call:
## lm(formula = card ~ ., data = stuDataTrain)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.9853 -0.4785  0.1833  0.2878  0.6123
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.349e-01  1.437e-02  37.221  < 2e-16 ***
## age         -1.700e-03  3.789e-04  -4.487 7.28e-06 ***
## income       2.098e-02  2.524e-03   8.311  < 2e-16 ***
## share        1.544e+00  7.474e-02  20.663  < 2e-16 ***
## expenditure -5.188e-06  2.726e-05  -0.190    0.849
## owner        1.024e-01  7.450e-03  13.747  < 2e-16 ***
## selfemp     -9.076e-02  1.275e-02  -7.119 1.13e-12 ***
## dependents  -1.252e-02  2.782e-03  -4.501 6.81e-06 ***
## months       3.889e-05  5.358e-05   0.726    0.468
## majorcards   6.270e-02  8.318e-03   7.538 5.04e-14 ***
## active       2.210e-03  5.345e-04   4.134 3.59e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3994 on 15789 degrees of freedom
## Multiple R-squared:  0.1502, Adjusted R-squared:  0.1496
## F-statistic:   279 on 10 and 15789 DF,  p-value: < 2.2e-16
```

```
getStuRMSE = function(thismodel){
  mean((predict(thismodel,stuDataValid)-stuDataValid$card)^2)^0.5
} # BIC give variables that are might right
getStuRMSE(basicLM)
```

```
## [1] 0.3959001
```

```
getStuRMSE(lm(card~.^2, data = stuDataTrain)) # 0.2880693
```

```
## [1] 0.286194
```

```
getStuRMSE(lm(card~.-owner, data = stuDataTrain))
```

```
## [1] 0.397857
```

```
getStuRMSE(lm(card~.-selfemp, data = stuDataTrain))
```

```
## [1] 0.3959549
```

```
getStuRMSE(lm(card~.-dependents, data = stuDataTrain)) #0.3988295
```

```
## [1] 0.3964949
```

```
getStuRMSE(lm(card~.+poly(income,3), data = stuDataTrain))
```

```
## Warning in predict.lm(thismodel, stuDataValid): prediction from a rank-deficient
## fit may be misleading
```

```
## [1] 0.3948921
```

```
getStuRMSE(lm(card~.+poly(expenditure,3), data = stuDataTrain)) #0.3535913
```

```
## Warning in predict.lm(thismodel, stuDataValid): prediction from a rank-deficient
## fit may be misleading
```

```
## [1] 0.3566737
```

```
getStuRMSE(lm(card~.+poly(active,3), data = stuDataTrain))
```

```
## Warning in predict.lm(thismodel, stuDataValid): prediction from a rank-deficient
## fit may be misleading
```

```
## [1] 0.3935132
```

```
getStuRMSE(lm(card~.^5, data = stuDataTrain)) # 0.2707599
```

```
## Warning in predict.lm(thismodel, stuDataValid): prediction from a rank-deficient
## fit may be misleading
```

```
## [1] 0.2715584
```

```
# earth
library(earth)
```

```
## Warning: package 'earth' was built under R version 4.1.2
```

```
## Loading required package: Formula
```

```
## Loading required package: plotmo
```

```
## Loading required package: plotrix
```

```
## Loading required package: TeachingDemos
```

```
getStuRMSE(earth(card~., data = stuDataTrain))
```

```
## [1] 0.2563334
```

```
getStuRMSE(earth(card~., data = stuDataTrain,degree = 2, thres = .00001))
```

```
## [1] 0.2296511
```

```
getStuRMSE(earth(card~., data = stuDataTrain,degree = 2, thres = .01)) # less thres more complex
```

```
## [1] 0.2307017
```

```
# RandomForest
install.packages("randomForest") #regression tree?  random forest take regression tree avg
```

```
## Installing package into 'C:/Users/92998/OneDrive/Documents/R/win-library/4.1'
## (as 'lib' is unspecified)
```

```
## package 'randomForest' successfully unpacked and MD5 sums checked
```

```
## Warning: cannot remove prior installation of package 'randomForest'
```

```
## Warning in file.copy(savedcopy, lib, recursive = TRUE):
## problem copying C:\Users\92998\OneDrive\Documents\R\win-
## library\4.1\00LOCK\randomForest\libs\x64\randomForest.dll
## to C:\Users\92998\OneDrive\Documents\R\win-
## library\4.1\randomForest\libs\x64\randomForest.dll: Permission denied
```

```
## Warning: restored 'randomForest'
```

```
##
## The downloaded binary packages are in
##  C:\Users\92998\AppData\Local\Temp\RtmpK0xXUr\downloaded_packages
```

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.1.2
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
getStuRMSE(randomForest(card~., data = stuDataTrain))
```

```
## Warning in randomForest.default(m, y, ...): The response has five or fewer
## unique values. Are you sure you want to do regression?
```

```
## [1] 0.2362053
```

```
########################
# Implementing k-fold Cross Validation
########################


getBankDataKFoldRMSE <- function(testFit){
  set.seed(3456987)
  totalFold = 10
  foldNum <- floor(runif(nrow(stuData)) * totalFold) + 1

  thisModelRMSE <- rep(NA, totalFold)
  for (thisFold in 1:totalFold) {
  trainingData <- subset(stuData, foldNum!=thisFold)
  validationData <- subset(stuData, foldNum == thisFold)
  thisModel <- update(testFit, data = trainingData)
  # update: refit: becuase we run ten times
  thisFit <- mean((predict(thisModel, validationData)-validationData$card)^2)^0.5
  thisModelRMSE[thisFold] = thisFit
  }
  return(mean(thisModelRMSE))
}

getBankDataKFoldRMSE(lm(card~., data = stuData))
```

```
## [1] 0.3987211
```

```
getBankDataKFoldRMSE(lm(card~.-majorcards,data = stuData))
```

```
## [1] 0.3995204
```

```
getBankDataKFoldRMSE(lm(card~.-selfemp, data = stuData))
```

```
## [1] 0.3992348
```

```
getBankDataKFoldRMSE(lm(card~.-dependents, data = stuData))
```

```
## [1] 0.3990389
```

```
getBankDataKFoldRMSE(lm(card~.-months, data = stuData))
```

```
## [1] 0.3987038
```

```
getBankDataKFoldRMSE(lm(card~.-age,data = stuData))
```

```
## [1] 0.3989376
```

```
getBankDataKFoldRMSE(lm(card~.+poly(income,5),data = stuData))
```

```
## Warning in predict.lm(thisModel, validationData): prediction from a rank-
## deficient fit may be misleading
```

```
## Warning in predict.lm(thisModel, validationData): prediction from a rank-
## deficient fit may be misleading

## Warning in predict.lm(thisModel, validationData): prediction from a rank-
## deficient fit may be misleading

## Warning in predict.lm(thisModel, validationData): prediction from a rank-
## deficient fit may be misleading

## Warning in predict.lm(thisModel, validationData): prediction from a rank-
## deficient fit may be misleading

## Warning in predict.lm(thisModel, validationData): prediction from a rank-
## deficient fit may be misleading

## Warning in predict.lm(thisModel, validationData): prediction from a rank-
## deficient fit may be misleading

## Warning in predict.lm(thisModel, validationData): prediction from a rank-
## deficient fit may be misleading

## Warning in predict.lm(thisModel, validationData): prediction from a rank-
## deficient fit may be misleading

## Warning in predict.lm(thisModel, validationData): prediction from a rank-
## deficient fit may be misleading
```

```
## [1] 0.3968683
```

```
getBankDataKFoldRMSE(earth(card~.,data = stuData, degree = 2, thres = 0.00001))
```

```
## [1] 0.2355741
```

```
# overfit: one model fits well does not fit well in out
# dataset

# Best Model
model1A = earth(card~.,data = stuData, degree = 2, thres = 0.00001)
```

Question 1B:

```
#TRAIN AND ESTIMATE YOUR SECOND MODEL (WITH REPORTS) HERE

stuData_B = read.csv('C:\\Users\\92998\\OneDrive\\Desktop\\Simon\\Predictive\\Student_Data_3.csv')
isTraining1 <- runif(nrow(stuData_B)) < .8
stuDataTrain1 <- subset(stuData_B, isTraining1)
stuDataValid1 <- subset(stuData_B, !isTraining1)
basicLM2 <- lm(card~., data = stuDataTrain1)
summary(basicLM2)
```

```
##
## Call:
## lm(formula = card ~ ., data = stuDataTrain1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.8853 -0.1655  0.1506  0.2476  0.9515
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.623e-01  1.337e-02  42.048  < 2e-16 ***
## reports     -1.215e-01  2.314e-03 -52.495  < 2e-16 ***
## age         -1.345e-03  3.494e-04  -3.848 0.000119 ***
## income       2.343e-02  2.318e-03  10.108  < 2e-16 ***
## share        1.440e+00  6.938e-02  20.762  < 2e-16 ***
## expenditure -4.326e-05  2.506e-05  -1.727 0.084277 .
## owner        5.079e-02  6.896e-03   7.365 1.86e-13 ***
## selfemp     -7.821e-02  1.160e-02  -6.745 1.58e-11 ***
## dependents  -1.120e-02  2.566e-03  -4.364 1.28e-05 ***
## months       1.053e-04  5.009e-05   2.103 0.035489 *
## majorcards   5.908e-02  7.687e-03   7.685 1.62e-14 ***
## active       8.142e-03  5.025e-04  16.204  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3679 on 15829 degrees of freedom
## Multiple R-squared:  0.2803, Adjusted R-squared:  0.2798
## F-statistic: 560.5 on 11 and 15829 DF,  p-value: < 2.2e-16
```

```
getStuRMSE2 = function(thismodel){
  mean((predict(thismodel,stuDataValid1)-stuDataValid1$card)^2)^0.5
}
getStuRMSE2(basicLM2)
```

```
## [1] 0.3674493
```

```
getStuRMSE2(lm(card~.^2, data = stuDataTrain1))
```

```
## [1] 0.2786645
```

```
getStuRMSE2(lm(card~.-owner, data = stuDataTrain1))
```

```
## [1] 0.3686117
```

```
getStuRMSE2(lm(card~.-selfemp, data = stuDataTrain1))
```

```
## [1] 0.3681218
```

```
getStuRMSE2(lm(card~.-dependents, data = stuDataTrain1))
```

```
## [1] 0.3676447
```

```
getStuRMSE2(lm(card~.+poly(income,3), data = stuDataTrain1))
```

```
## Warning in predict.lm(thismodel, stuDataValid1): prediction from a rank-
## deficient fit may be misleading
```

```
## [1] 0.3660438
```

```
getStuRMSE2(lm(card~.+poly(expenditure,3), data = stuDataTrain1))
```

```
## Warning in predict.lm(thismodel, stuDataValid1): prediction from a rank-
## deficient fit may be misleading
```

```
## [1] 0.3363142
```

```
getStuRMSE2(lm(card~.+poly(active,3), data = stuDataTrain1))
```

```
## Warning in predict.lm(thismodel, stuDataValid1): prediction from a rank-
## deficient fit may be misleading
```

```
## [1] 0.3633692
```

```
getStuRMSE2(lm(card~.^5, data = stuDataTrain1)) # lead # 0.2610449
```

```
## Warning in predict.lm(thismodel, stuDataValid1): prediction from a rank-
## deficient fit may be misleading
```

```
## [1] 0.2610449
```

```
getBankDataKFoldRMSE2 <- function(testFit){
  set.seed(3456987)
  totalFold = 10
  foldNum <- floor(runif(nrow(stuData_B)) * totalFold) + 1

  thisModelRMSE <- rep(NA, totalFold)
  for (thisFold in 1:totalFold) {
  trainingData <- subset(stuData_B, foldNum!=thisFold)
  validationData <- subset(stuData_B, foldNum == thisFold)
  thisModel <- update(testFit, data = trainingData)
  # update: refit: becuase we run ten times
  thisFit <- mean((predict(thisModel, validationData)-validationData$card)^2)^0.5
  thisModelRMSE[thisFold] = thisFit
  }
  return(mean(thisModelRMSE))
}

getBankDataKFoldRMSE2(lm(card~., data = stuData_B))
```

```
## [1] 0.3678695
```

```
getBankDataKFoldRMSE2(lm(card~.-majorcards,data = stuData_B))
```

```
## [1] 0.3685638
```

```
getBankDataKFoldRMSE2(lm(card~.-selfemp, data = stuData_B))
```

```
## [1] 0.3684125
```

```
getBankDataKFoldRMSE2(lm(card~.-dependents, data = stuData_B))
```

```
## [1] 0.3680698
```

```
getBankDataKFoldRMSE2(lm(card~.-months, data = stuData_B))
```

```
## [1] 0.3679012
```

```
getBankDataKFoldRMSE2(lm(card~.-age,data = stuData_B))
```

```
## [1] 0.3680379
```

```
getBankDataKFoldRMSE2(lm(card~.+poly(income,5),data = stuData_B))
```

```
## Warning in predict.lm(thisModel, validationData): prediction from a rank-
## deficient fit may be misleading
```

```
## Warning in predict.lm(thisModel, validationData): prediction from a rank-
## deficient fit may be misleading

## Warning in predict.lm(thisModel, validationData): prediction from a rank-
## deficient fit may be misleading

## Warning in predict.lm(thisModel, validationData): prediction from a rank-
## deficient fit may be misleading

## Warning in predict.lm(thisModel, validationData): prediction from a rank-
## deficient fit may be misleading

## Warning in predict.lm(thisModel, validationData): prediction from a rank-
## deficient fit may be misleading

## Warning in predict.lm(thisModel, validationData): prediction from a rank-
## deficient fit may be misleading

## Warning in predict.lm(thisModel, validationData): prediction from a rank-
## deficient fit may be misleading

## Warning in predict.lm(thisModel, validationData): prediction from a rank-
## deficient fit may be misleading

## Warning in predict.lm(thisModel, validationData): prediction from a rank-
## deficient fit may be misleading
```

```
## [1] 0.3658327
```

```
getBankDataKFoldRMSE2(earth(card~.,data = stuData_B, degree = 2, thres = 0.00001))
```

```
## [1] 0.2342423
```

```
model1B <- earth(card~.,data = stuData_B, degree = 2, thres = 0.00001)
```

```
## [1] "MyModels.Rdata generated!  Please submit this file via blackboard."
```