Case 5.4 Build Regression using Regression Node (Using R)

5.4.1 Purpose of the Study

To understand how to build a regression model using R.

5.4.2 Data

The data set used in this study is the same data used in Association for Computing Machinery's (ACM) 1998 KDD-Cup competition (http://kdd.ics.uci.edu). The original data is much more complicate and the version used is here from SAS Institute modified version. The response rate for this donation campaign is about 5% although you can see the response rate in this subsample is 50%. This means the prior probability is 0.05.

```
require(readxl)
cup98LRN <- as.data.frame(read_excel( "F:\\R\\pmad_pva.xlsx"
))
cup98LRN <- cup98LRN[!is.na(cup98LRN$TargetD),]
```

A great aspect of the caret package is that it handles a good portion of data preparation for us. Here we use the (preProcess) function to

- remove zero or near zero variance columns
- remove columns that are highly correlated with other columns
- Box-Cox transform the columns where appropriate
- center and scale the columns to have mean zero variance one
- impute missing values with median values
- apply the spatial sign transform

```
y <- cup98LRN$TargetD
X <- cup98LRN[,-c(1:3)]
require(caret)
```

```
preProcess.X <- preProcess(X, method = c(
    "zv","nzv","corr", "BoxCox", "center","scale", "medianImpute", "spatialSign"
  ),
  outcome = y
)
X <- predict(preProcess.X,X)
```

There are several categorical variables. Here we bin each categorical vector into two groups; one group is a bin of the levels with smaller conditional averages of TargetD, the other group is a bin of the levels with larger conditional averages of TargetD. The bins were constructed so that the smaller/larger partition gives them a split as close to 50-50 as possible. The bins are represented as binary numeric vectors. Notice that after binning all columns are numeric vectors; many model functions complete their computations faster with numeric variables.

```
X$StatusCat96NK <- as.numeric(X$StatusCat96NK == "S") BIN.DemCluster <- c(
  "04","08","09","16","19","20","23","25","26","27",
  "28","30","32","33","36","38","39","40","41","43",
  "45","46","47","48","49","51","52","53"
)
X$DemCluster <- as.numeric(X$DemCluster %in% BIN.DemCluster) X$DemGender <-
as.numeric(X$DemGender == "F")
X$DemHomeOwner <- as.numeric(X$DemHomeOwner == "H")
```

5.4.3 Regression Modeling in R

```
list.model <- list() list.model[["lm"]] <- train(
  y = y, x = X,
  trControl = trainControl(method = "repeatedcv",number = 5,repeats = 5), method = "lm"
)
for(j in c("forward","both","backward")){ list.model[[j]]  <-  train(
    y = y, x = X,
    trControl = trainControl(method = "repeatedcv",number = 5,repeats = 5), method =
    "lmStepAIC",
    trace = 0, direction = j
  )
}
```

Lets look at the coefficients.

```
sapply(X = list.model,FUN = function(x) coef(x$finalModel))

## $lm
##            (Intercept)              GiftCnt36        GiftCntCard36      GiftCntCardAll
```

```
##        17.7457247      1.7967810    -1.4469884    -1.3292366
##       GiftAvgLast     GiftAvg36     GiftAvgAll    GiftTimeLast
##        31.1333606     23.8100025    -3.8781869     0.6333394
##     GiftTimeFirst      PromCnt12     PromCnt36     PromCntAll
##        -4.2509375      3.0695342    -4.7773528     3.2241325
##     PromCntCard12   PromCntCard36  StatusCat96NK StatusCatStarAll
##        -0.8038078      1.4134616    -0.5439398     0.5632963
##        DemCluster         DemAge     DemGender    DemHomeOwner
##        -0.6793548      0.3396813    -0.6120642    -0.6880468
##   DemMedHomeValue DemPctVeterans   DemMedIncome
##        -0.2974096      0.7663830     0.1339993
##
## $forward
##       (Intercept)      GiftCnt36  GiftCntCard36   GiftCntCardAll
##        17.7457247      1.7967810    -1.4469884    -1.3292366
##       GiftAvgLast      GiftAvg36     GiftAvgAll    GiftTimeLast
##        31.1333606     23.8100025    -3.8781869     0.6333394
##     GiftTimeFirst      PromCnt12      PromCnt36     PromCntAll
##        -4.2509375      3.0695342    -4.7773528     3.2241325
##     PromCntCard12  PromCntCard36  StatusCat96NK StatusCatStarAll
##        -0.8038078      1.4134616    -0.5439398     0.5632963
##        DemCluster         DemAge      DemGender    DemHomeOwner
##        -0.6793548      0.3396813    -0.6120642    -0.6880468
##   DemMedHomeValue DemPctVeterans   DemMedIncome
##        -0.2974096      0.7663830     0.1339993
##
## $both
##    (Intercept)     GiftCnt36 GiftCntCard36   GiftAvgLast     GiftAvg36
##     17.7660595     1.6435803    -1.6973733    31.1680536    23.6571237
##     GiftAvgAll GiftTimeFirst     PromCnt12     PromCnt36    PromCntAll
##     -3.7508644    -4.5883317     2.0617651    -4.2861356     2.9553127
##  PromCntCard36 StatusCat96NK    DemCluster     DemGender  DemHomeOwner
##      1.2345969    -0.6015682    -0.6397340    -0.6221611    -0.6837266
## DemPctVeterans
##      0.8079780
##
## $backward
```

```
##    (Intercept)     GiftCnt36 GiftCntCard36   GiftAvgLast     GiftAvg36
##     17.7660595     1.6435803    -1.6973733    31.1680536    23.6571237
##     GiftAvgAll GiftTimeFirst     PromCnt12     PromCnt36    PromCntAll
##     -3.7508644    -4.5883317     2.0617651    -4.2861356     2.9553127
##  PromCntCard36 StatusCat96NK    DemCluster     DemGender  DemHomeOwner
##      1.2345969    -0.6015682    -0.6397340    -0.6221611    -0.6837266
## DemPctVeterans
##      0.8079780
```

Now let’s look at cross-validation statistics.

Case 5.4 Build Regression using Regression Node (Using R)

```
sapply(X  =  list.model,FUN  =  getTrainPerf)

##       lm        forward both      backward
## TrainRMSE 9.026997          9.01431          9.051548         9.065909
## TrainRsquared 0.4738884 0.4755433           0.4718298        0.4699284
## TrainMAE    4.691627          4.693219         4.697055         4.698826
## method        "lm"       "lmStepAIC" "lmStepAIC" "lmStepAIC"
```

There appears to be two models, the full model and a reduced model. Let's run an ANOVA test to see if the full model is sufficiently better than the reduced model.

```
reduced.model <- list.model[["backward"]]$finalModel
full.model  <-  list.model[["lm"]]$finalModel anova(reduced.model,full.model)

## Analysis of Variance Table
##
## Model 1: .outcome ~ GiftCnt36 + GiftCntCard36 + GiftAvgLast + GiftAvg36 +
##      GiftAvgAll + GiftTimeFirst + PromCnt12 + PromCnt36 + PromCntAll +
##      PromCntCard36 + StatusCat96NK + DemCluster + DemGender +
##      DemHomeOwner + DemPctVeterans
## Model 2: .outcome ~ GiftCnt36 + GiftCntCard36 + GiftCntCardAll + GiftAvgLast +

##      GiftAvg36 + GiftAvgAll + GiftTimeLast + GiftTimeFirst + PromCnt12 +
##      PromCnt36 + PromCntAll + PromCntCard12 + PromCntCard36 +
##      StatusCat96NK + StatusCatStarAll + DemCluster + DemAge + DemGender +
##      DemHomeOwner + DemMedHomeValue + DemPctVeterans +
##      DemMedIncome
##    Res.Df       RSS Df Sum of Sq        F  Pr(>F)
## 1     4827 398497
## 2     4820 398192             305.32 0.528    0.814
         7
```

It looks like the reduced model is just as good as the full model. Since they are equally good, the simpler one is better; let's check it out.

Case 5.4 Build Regression using Regression Node (Using R)

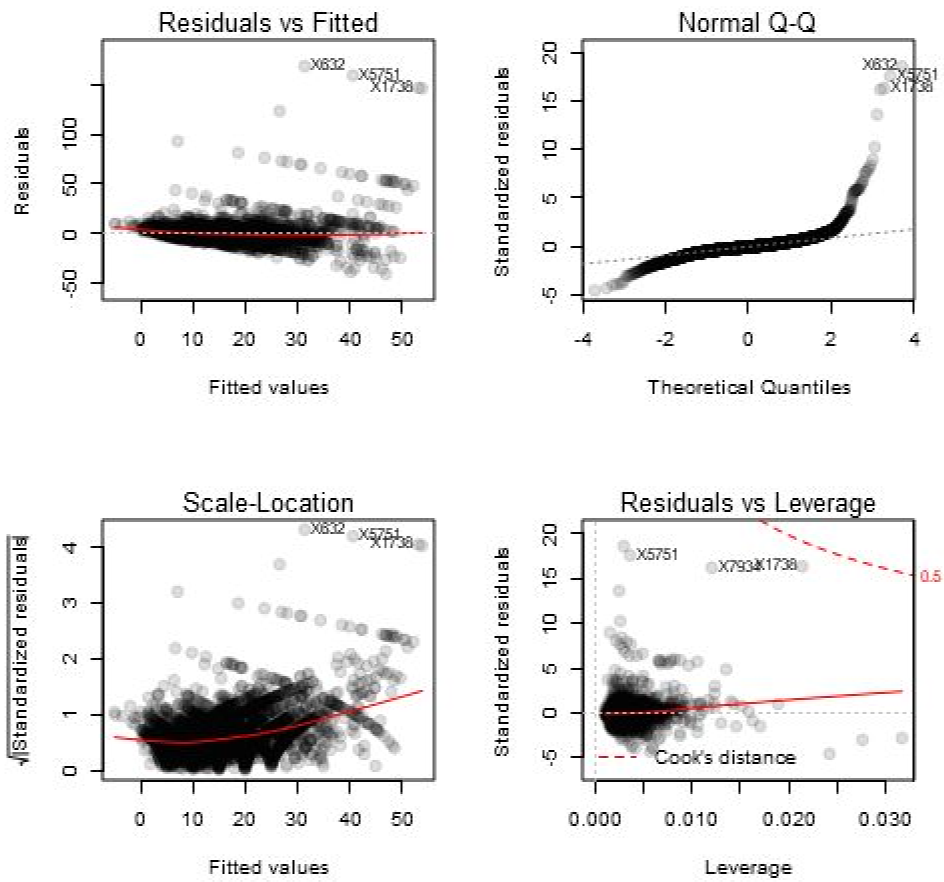
```
summary(reduced.model)
##
#   Call:
#   lm(formula = .outcome ~ GiftCnt36 + GiftCntCard36 + GiftAvgLast +
#        GiftAvg36 + GiftAvgAll + GiftTimeFirst + PromCnt12 + PromCnt36     +
#        PromCntAll + PromCntCard36 + StatusCat96NK    + DemCluster  +
#        DemGender   DemHomeOwner  +                   data  = dat)
#        +              DemPctVeterans,
#    Residuals:
#        Min          1Q     Median           3Q         Max
#   -40.928    -3.099     -0.647          2.383 168.587
#
#
#
#
#
#
#
#
#
#
#
#   Signif.    codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
#   Residual standard error: 9.086 on 4827 degrees of freedom
#   Multiple  R-squared:      0.4686,Adjusted  R-squared:       0.467
#   F-statistic: 283.8 on 15 and 4827 DF,      p-value:           < 2.2e-16
#
anova(reduced.model)
#
#   Analysis of Variance Table
#
#   Response   .outcome
#   :                        Df     Sum      Mean    F  value          Pr(>F)
#                             1      Sq        Sq   816.0332  <  2.2e-16 ***
#   GiftCnt36                      67368     67368
#   GiftCntCard36             1                      0.0135     0.90750
#   GiftAvgLast               1        1         1  3147.748  <  2.2e-16 ***
#                                  25986   259865        5
#                                      5
```

```
## Coefficients:
##                 Estimate  Std. Error  t value  Pr(>|t|)
## (Intercept)      17.7661      0.3027   58.690   < 2e-16 ***
## GiftCnt36         1.6436      1.0340    1.589  0.112016
## GiftCntCard36    -1.6974      0.9727   -1.745  0.081053 .
## GiftAvgLast      31.1681      1.6170   19.275   < 2e-16 ***
## GiftAvg36        23.6571      1.6041   14.748   < 2e-16 ***
## GiftAvgAll       -3.7509      1.0819   -3.467  0.000531 ***
## GiftTimeFirst    -4.5883      1.6061   -2.857  0.004296 **
## PromCnt12         2.0618      1.0793    1.910  0.056165 .
## PromCnt36        -4.2861      1.5779   -2.716  0.006624 **
## PromCntAll        2.9553      2.0924    1.412  0.157896
## PromCntCard36     1.2346      0.8348    1.479  0.139246
## StatusCat96NK    -0.6016      0.3681   -1.634  0.102276
## DemCluster       -0.6397      0.2693   -2.376  0.017545 *
## DemGender        -0.6222      0.2625   -2.370  0.017839 *
## DemHomeOwner     -0.6837      0.2651   -2.580  0.009922 **
## DemPctVeterans    0.8080      0.5125    1.577  0.114944
## ---
```

```
## GiftAvg36         1  18867   18867  228.5421  < 2.2e-16  ***
## GiftAvgAll        1     30      30    0.3663   0.54503
## GiftTimeFirst     1   3028    3028   36.6813  1.497e-09  ***
## PromCnt12         1     28      28    0.3439   0.55759
## PromCnt36         1    293     293    3.5517   0.05954  .
                     1
```

```
## Signif. codes:    0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 par(mfrow = c(2,2))
plot(reduced.model,pch = 19,col = "#00000022")
```

```
## PromCntAll              97      97    1.1734   0.27876
## PromCntCard36     1    144     144    1.7502   0.18591
## StatusCat96NK     1    218     218    2.6377   0.10442
## DemCluster        1    336     336    4.0733   0.04362  *
## DemGender         1    460     460    5.5690   0.01832  *
## DemHomeOwne       1    497     497    6.0163   0.01421  *
   r
## DemPctVeterans    1    205     205    2.4858   0.11494
## Residuals      4827 398497      83
## ---
```




```
dev.off()

## windows
##       2
```

Appendix 1 Discussion Questions

What are the optimal parameter settings for entry into a forward selection model?

Case 5.4 Build Regression using Regression Node (Using R)

1. What are the optimal parameter settings for exit from a backward elimination model?
2. Why we prefer to use validation error on selecting the best model?