2018R2 Data Mining (STAT5104) Assignment 3

Yiu Chung WONG 1155017920

```
knitr::opts_chunk$set(echo = TRUE)
library(dplyr);
library(ROCR);
library(ggplot2);
set.seed(17920);
```

Q1

a)

```
d <- read.csv("tele.csv", header = TRUE, sep = ","); #read data
inTrain <- caret::createDataPartition(d$Change,     #create index for train / test partition
                                      p = .8,
                                      list = FALSE);

d0 <- d[inTrain,];                                   #select observations for training
d1 <- d[-inTrain,];                                  #select observations for testing
```

b)

```
y0 <- factor(d0[,18],
                 labels = c("stay", "change"),
                 levels = c(0, 1));
y1 <- factor(d1[,18],
                 labels = c("stay", "change"),
                 levels = c(0, 1));

continuous <- c(5, 7, 8, 10, 11, 13, 14, 16);      #Column index for continuous variables
x0 <- d0[, continuous];                             #Select continuous variables for training data
x1 <- d1[, continuous];                             #Select continuous variables for testing data

fit <- glm(y0 ~ .,
           data=x0,
           family = binomial(link='logit'));       #Logistic Regression

tel.lreg <- fit %>%
            MASS::stepAIC(.,
                          trace = FALSE,
                          direction = "backward");
```

c)

```r
train.probabilities <- predict(tel.lreg ,type='response');
train.predicted.classes <- ifelse(train.probabilities > 0.5, 1, 0) %>%
  factor(.,
         labels = c("stay", "change"),
         levels = c(0, 1));
train.classification.table <- table(y0, train.predicted.classes);
train.err.rate <- (train.classification.table[1,2] +
                     train.classification.table[2,1]) / sum(train.classification.table);
train.classification.table;
```

```
##         train.predicted.classes
## y0       stay change
##   stay   2264      0
##   change  361      6
```

```r
test.probabilities <- predict(tel.lreg, newdata = x1 ,type='response');
test.predicted.classes <- ifelse(test.probabilities > 0.5, 1, 0) %>%
  factor(.,
         labels = c("stay", "change"),
         levels = c(0, 1));
test.classification.table <- table(y1, test.predicted.classes);
test.err.rate <- (test.classification.table[1,2] +
                    test.classification.table[2,1]) / sum(test.classification.table);
q1.err.rate = test.err.rate;
test.classification.table;
```

```
##         test.predicted.classes
## y1       stay change
##   stay    549      0
##   change  105      3
```

The train error rate is (0 + 361) / 2631 = 0.1372102

The test error rate is (0 + 105) / 657 = 0.1598174

Q2

a)

```r
x0 <- d0[, continuous];                               #Select continuous variables for training data
x1 <- d1[, continuous];                               #Select continuous variables for testing data
y0 <- d0[,18];
y1 <- d1[,18];                                        #label
```

b)

```r
tele.nn = ann(x = x0, y = y0, size = 7, linout = T, try = 30);
```

the best (smallest) objective function value among 30 trials is 225.9820308

```r
train.predicted.classes <- round(tele.nn$fitted.values) %>%
  factor(.,
         labels = c("stay", "change"),
         levels = c(0, 1));
train.classification.table <- table(y0, train.predicted.classes);
train.err.rate <- (train.classification.table[1,2] +
                     train.classification.table[2,1]) / sum(train.classification.table);
train.classification.table;
```

```
##    train.predicted.classes
## y0   stay change
##    0 2218     46
##    1  211    156
```

```r
test.predicted.classes <- predict(tele.nn, newdata = x1) %>%
  round(.) %>%
  factor(.,
         labels = c("stay", "change"),
         levels = c(0, 1));
test.classification.table <- table(y1, test.predicted.classes);
test.err.rate <- (test.classification.table[1,2] +
                    test.classification.table[2,1]) / sum(test.classification.table);
test.classification.table;
```

```
##    test.predicted.classes
## y1   stay change
##    0  531     18
##    1   65     43
```

The train error rate is (46 + 211) / 2631 = 0.0976815

The test error rate is (18 + 65) / 657 = 0.1263318

c)

```
ann.test.err.rates <- c(test.err.rate);
for (i in 8:9)
{
  tele.nn = ann(x = x0, y = y0, size = i, linout = T, try = 30);
  test.predicted.classes <- predict(tele.nn, newdata = x1) %>%
  round(.) %>%
  factor(.,
         labels = c("stay", "change"),
         levels = c(0, 1));
  test.classification.table <- table(y1, test.predicted.classes);
  test.err.rate <- (test.classification.table[1,2] +
                      test.classification.table[2,1]) / sum(test.classification.table);
  ann.test.err.rates = c(ann.test.err.rates, test.err.rate);
}

names(ann.test.err.rates) <- c("Size = 7", "Size = 8", "Size = 9");
```

The lowest test error rate among size = 7, 8, 9 is when size == 8 with rate 0.1232877 which is slightly lower then the one in Q1 (0.1598174).

Q3

a)

```
x <- d[, continuous];           #Extract all the continuous variables in d
z <- stand(x);
```

b)

```
tel.km2 <- km(x = z, try = 10, k = 2);
```

```
## cluster size= 1664 1624
## stat= 648.2585
```

```
tel.km3 <- km(x = z, try = 10, k = 3);
```

```
## cluster size= 1085 1090 1113
## stat= 633.0744
```

```
tel.km4 <- km(x = z, try = 10, k = 4);
```

```
## cluster size= 842 858 775 813
## stat= 627.2107
```

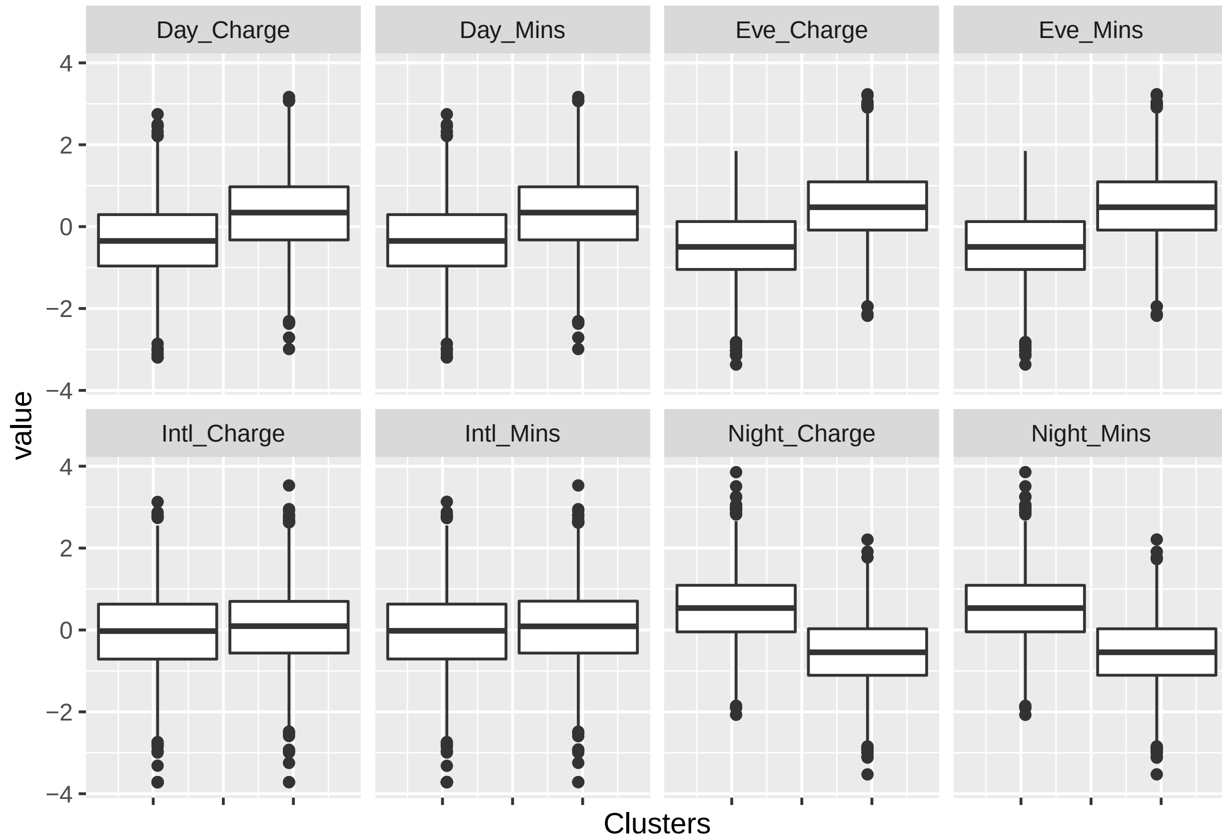
```
tel.km5 <- km(x = z, try = 10, k = 5);
```

```
## cluster size= 657 680 604 674 673
## stat= 634.76
```

K == 2 gives the highest R statistics

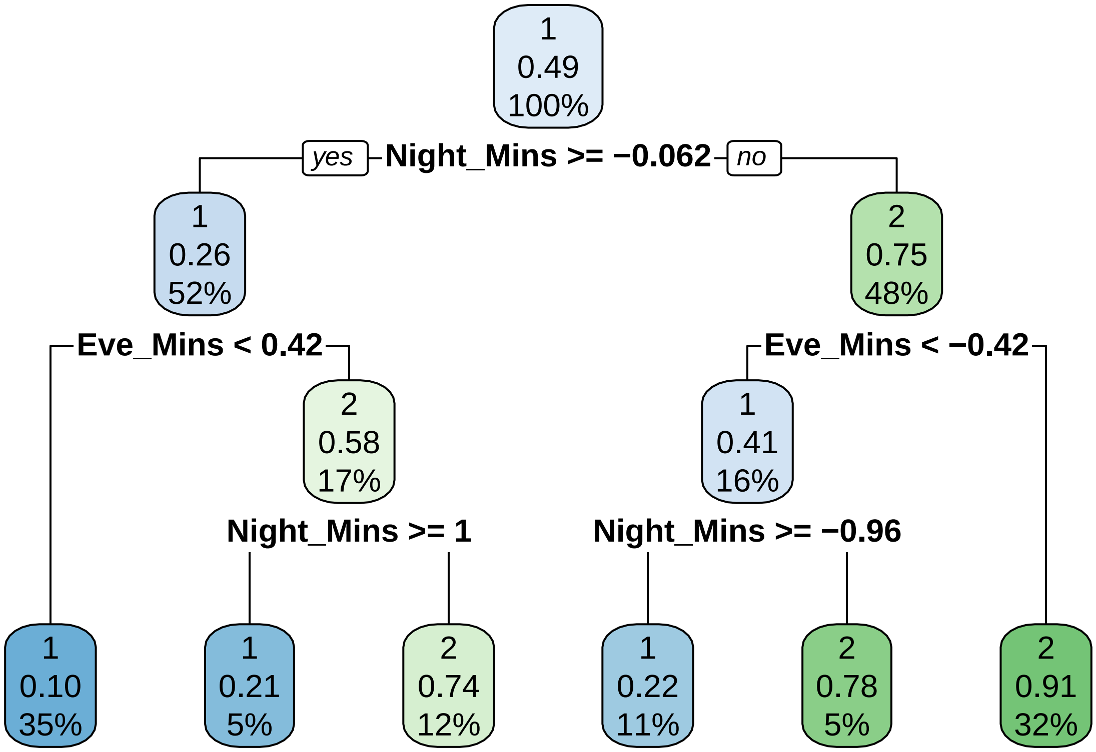
c)

```
comp.df <- cbind(z, tel.km2);
names(comp.df)[ncol(comp.df)] <- "cluster";
comp.df.long <- tidyr::gather(comp.df, key, value, -cluster);
ggplot(comp.df.long,
       aes(group = cluster,
           y = value)) +
  geom_boxplot() +
  theme(axis.text.x = element_blank()) +
  facet_wrap(.~key, nrow=2) +
  xlab("Clusters");
```




```
control <- rpart::rpart.control(maxdepth = 3);
tel.ctree <- rpart::rpart(data = z,
                          formula = tel.km2 ~ .,
                          control = control,
                          method = 'class');

rpart.plot::rpart.plot(tel.ctree);         #print plot
```




```
print(tel.ctree);
```

```
## n= 3288
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 3288 1624 1 (0.50608273 0.49391727)
##    2) Night_Mins>=-0.06245751 1708  446 1 (0.73887588 0.26112412)
##      4) Eve_Mins< 0.4243272 1142  115 1 (0.89929947 0.10070053) *
##      5) Eve_Mins>=0.4243272 566  235 2 (0.41519435 0.58480565)
##       10) Night_Mins>=1.017686 168   36 1 (0.78571429 0.21428571) *
##       11) Night_Mins< 1.017686 398  103 2 (0.25879397 0.74120603) *
##    3) Night_Mins< -0.06245751 1580  402 2 (0.25443038 0.74556962)
##      6) Eve_Mins< -0.4167671 529  218 1 (0.58790170 0.41209830)
##       12) Night_Mins>=-0.9639008 350   78 1 (0.77714286 0.22285714) *
##       13) Night_Mins< -0.9639008 179   39 2 (0.21787709 0.78212291) *
##      7) Eve_Mins>=-0.4167671 1051   91 2 (0.08658421 0.91341579) *
```

```
rattle::asRules(tel.ctree, compact=FALSE); #print rules
```

```
##
##  Rule number: 7 [tel.km2=2 cover=1051 (32%) prob=0.91]
##    Night_Mins< -0.06246
```

```
##    Eve_Mins>=-0.4168
##
##  Rule number: 13 [tel.km2=2 cover=179 (5%) prob=0.78]
##    Night_Mins< -0.06246
##    Eve_Mins< -0.4168
##    Night_Mins< -0.9639
##
##  Rule number: 11 [tel.km2=2 cover=398 (12%) prob=0.74]
##    Night_Mins>=-0.06246
##    Eve_Mins>=0.4243
##    Night_Mins< 1.018
##
##  Rule number: 12 [tel.km2=1 cover=350 (11%) prob=0.22]
##    Night_Mins< -0.06246
##    Eve_Mins< -0.4168
##    Night_Mins>=-0.9639
##
##  Rule number: 10 [tel.km2=1 cover=168 (5%) prob=0.21]
##    Night_Mins>=-0.06246
##    Eve_Mins>=0.4243
##    Night_Mins>=1.018
##
##  Rule number: 4 [tel.km2=1 cover=1142 (35%) prob=0.10]
##    Night_Mins>=-0.06246
##    Eve_Mins< 0.4243
```

d)

- Only `Night_Mins` and `Eve_Mins` are used to define classification rules in `tel.ctree`.