Intro to Data Science - HW 10


```
# Enter your name here: Bhavya Shah
```

Attribution statement: (choose only one and delete the rest)

```
# 1. I did this homework by myself, with help from the book and the professor.
```

Association mining can be applied to many data problems beyond the well-known example of **finding relationships between different products in customer shopping data**. In this homework assignment, we will explore **real data** from the banking sector and look for **patterns associated with the likelihood of responding positively to a direct marketing campaign and signing up for a term deposit with the bank (stored in the variable y)**.
You can find out more about the variables in this dataset here:
https://archive.ics.uci.edu/ml/datasets/bank+marketing (https://archive.ics.uci.edu/ml/datasets/bank+marketing)

Part 1: Explore Data Set

A. Read the contents of the following URL to a dataframe called **bank**
https://intro-datascience.s3.us-east-2.amazonaws.com/bank-full.csv (https://intro-datascience.s3.us-east-2.amazonaws.com/bank-full.csv)

Hint: Even though this is a .csv file, chances are R won t be able to read it in correctly using the read_csv() function. If you take a closer look at the contents of the URL file, you may notice each field is separated by a **semicolon** (;) rather than a comma.

In situations like this, consider using either **read.csv()** or **read.table()**, with two additional parameters. *sep=";"* defines how the data is seperated (the default is a comma), and *header=TRUE* defines that there is a header line in the dataset.

```
bank <- read.csv("https://intro-datascience.s3.us-east-2.amazonaws.com/bank-full.csv",
sep = ";", header = TRUE)
#reading the csv file with the data being separated by a ';' and with a header row
head(bank)
```

```
##   age       job marital   education default housing loan   contact month
## 1  56 housemaid married    basic.4y      no      no   no telephone   may
## 2  57  services married high.school unknown      no   no telephone   may
## 3  37  services married high.school      no     yes   no telephone   may
## 4  40    admin. married    basic.6y      no      no   no telephone   may
## 5  56  services married high.school      no      no  yes telephone   may
## 6  45  services married    basic.9y unknown      no   no telephone   may
##   day_of_week duration campaign pdays previous    poutcome emp.var.rate
## 1         mon      261        1   999        0 nonexistent          1.1
## 2         mon      149        1   999        0 nonexistent          1.1
## 3         mon      226        1   999        0 nonexistent          1.1
## 4         mon      151        1   999        0 nonexistent          1.1
## 5         mon      307        1   999        0 nonexistent          1.1
## 6         mon      198        1   999        0 nonexistent          1.1
##   cons.price.idx cons.conf.idx euribor3m nr.employed  y
## 1         93.994         -36.4     4.857        5191 no
## 2         93.994         -36.4     4.857        5191 no
## 3         93.994         -36.4     4.857        5191 no
## 4         93.994         -36.4     4.857        5191 no
## 5         93.994         -36.4     4.857        5191 no
## 6         93.994         -36.4     4.857        5191 no
```

B. Make sure there are **41,188** rows and **21** columns in your **bank** df.

```
dim(bank) #there are 41188 rows and 21 columns
```

```
## [1] 41188    21
```

C. Next, we will focus on some key factor variables from the dataset, and convert a few numeric ones to factor variables. Execute the following command. Write a comment describing how the conversion for each numeric variable works and what are the variables in the resulting dataframe.

```
bank_key <- data.frame(job=as.factor(bank$job),
                       marital=as.factor(bank$marital),
                       housing_loan=as.factor(bank$housing),
                       young=as.factor(bank$age<median(bank$age)),
                       contacted_more_than_once=as.factor(bank$campaign>1),
                       contacted_before_this_campaign=as.factor(bank$previous<0),
                       success=as.factor(bank$y))

#It transformed categorical data into numerical data.
#The job variable has been broken down into 12 factors.
#The marital variable is divided into four factors.
#The variable housing_loan is divided into three components.
#The young variable is calculated using the median age and divided into two parts.
#The contacted_more_than_once variable is transformed to a factor variable if the "campaign" var
iable is more than 1. The contacted_before_this_campaign variable is turned to a factor variable
if the "previous" variable is less than 0.
```

D. Count the number of successful term deposit sign-ups, using the table( ) command on the **success** variable.

```
success <- table(bank_key$success)
success
```

```
##
##    no   yes
## 36548  4640
```

```
#displays the number of term deposit sign-ups that were successful and those that were not
```

E. Express the results of problem C as percentages by sending the results of the table( ) command into the prop.table( ) command.

```
prop.table(success)*100
```

```
##
##       no      yes
## 88.73458 11.26542
```

```
#gives percentage value of successful and non successful term deposit sign ups
```

F. Using the same techniques, show the percentages for the **marital** and **housing_loan** variables as well.

```
martial <- table(bank_key$marital)
prop.table(martial)*100
```

```
##
##   divorced    married     single    unknown
## 11.1974361 60.5224823 28.0858502  0.1942313
```

```
housing_loan <- table(bank_key$housing_loan)
prop.table(housing_loan)*100
```

```
##
##        no   unknown       yes
## 45.212198  2.403613 52.384190
```

Part 2: Coerce the data frame into transactions

A. Install and library two packages: **arules** and **arulesViz**.

```
#install.packages("arules")
#install.packages("arulesViz")
library(arules)
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'arules'
```

```
## The following objects are masked from 'package:base':
##
##     abbreviate, write
```

```
library(arulesViz)
```

B. Coerce the **bank_new** dataframe into a **sparse transactions matrix** called **bankX**.

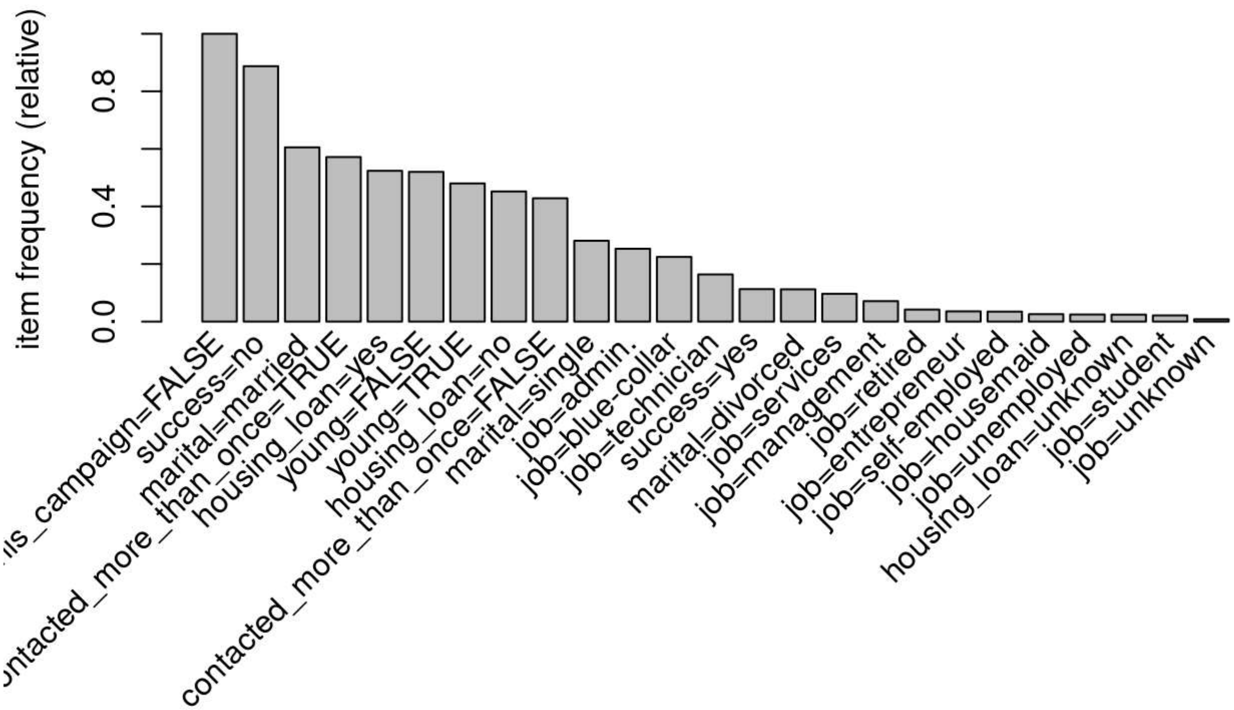
```
bankX <- as(bank_key, "transactions")
```

C. Use the itemFrequency( ) and itemFrequencyPlot( ) commands to explore the contents of **bankX**. What do you see?

```
itemFrequency(bankX)
```

```
##                      job=admin.                        job=blue-collar
##                     0.253034865                            0.224677090
##                job=entrepreneur                          job=housemaid
##                     0.035350102                            0.025735651
##                  job=management                            job=retired
##                     0.070991551                            0.041759736
##               job=self-employed                           job=services
##                     0.034500340                            0.096363018
##                     job=student                         job=technician
##                     0.021244052                            0.163712732
##                  job=unemployed                            job=unknown
##                     0.024618821                            0.008012042
##                marital=divorced                        marital=married
##                     0.111974361                            0.605224823
##                  marital=single                        marital=unknown
##                     0.280858502                            0.001942313
##                 housing_loan=no                   housing_loan=unknown
##                     0.452121977                            0.024036127
##                housing_loan=yes                            young=FALSE
##                     0.523841896                            0.520054385
##                      young=TRUE         contacted_more_than_once=FALSE
##                     0.479945615                            0.428328639
##   contacted_more_than_once=TRUE contacted_before_this_campaign=FALSE
##                     0.571671361                            1.000000000
##                      success=no                            success=yes
##                     0.887345829                            0.112654171
```

```
itemFrequencyPlot(bankX, topN=25)
```




```
#bar plot of most frequent items using the top 25 items
```

D. This is a fairly large dataset, so we will explore only the first 10 observations in the **bankX** transaction matrix:

```
inspect(bankX[1:10])
```

```
##      items                                    transactionID
## [1]  {job=housemaid,
##       marital=married,
##       housing_loan=no,
##       young=FALSE,
##       contacted_more_than_once=FALSE,
##       contacted_before_this_campaign=FALSE,
##       success=no}                                         1
## [2]  {job=services,
##       marital=married,
##       housing_loan=no,
##       young=FALSE,
##       contacted_more_than_once=FALSE,
##       contacted_before_this_campaign=FALSE,
##       success=no}                                         2
## [3]  {job=services,
##       marital=married,
##       housing_loan=yes,
##       young=TRUE,
##       contacted_more_than_once=FALSE,
##       contacted_before_this_campaign=FALSE,
##       success=no}                                         3
## [4]  {job=admin.,
##       marital=married,
##       housing_loan=no,
##       young=FALSE,
##       contacted_more_than_once=FALSE,
##       contacted_before_this_campaign=FALSE,
##       success=no}                                         4
## [5]  {job=services,
##       marital=married,
##       housing_loan=no,
##       young=FALSE,
##       contacted_more_than_once=FALSE,
##       contacted_before_this_campaign=FALSE,
##       success=no}                                         5
## [6]  {job=services,
##       marital=married,
##       housing_loan=no,
##       young=FALSE,
##       contacted_more_than_once=FALSE,
##       contacted_before_this_campaign=FALSE,
##       success=no}                                         6
## [7]  {job=admin.,
##       marital=married,
##       housing_loan=no,
##       young=FALSE,
##       contacted_more_than_once=FALSE,
##       contacted_before_this_campaign=FALSE,
##       success=no}                                         7
## [8]  {job=blue-collar,
##       marital=married,
```

```
##       housing_loan=no,
##       young=FALSE,
##       contacted_more_than_once=FALSE,
##       contacted_before_this_campaign=FALSE,
##       success=no}                                 8
## [9]  {job=technician,
##       marital=single,
##       housing_loan=yes,
##       young=TRUE,
##       contacted_more_than_once=FALSE,
##       contacted_before_this_campaign=FALSE,
##       success=no}                                 9
## [10] {job=services,
##       marital=single,
##       housing_loan=yes,
##       young=TRUE,
##       contacted_more_than_once=FALSE,
##       contacted_before_this_campaign=FALSE,
##       success=no}                                 10
```

E. Explain the difference between **bank_new** and **bankX** in a block comment:

```
#bank_new is a dataset containing the original variables of the 'bank' dataset that have been ch
anged into factors.

#bankX is a transaction object derived from 'bank_key'
#each row representing a transaction and each column representing an item.
```

Part 3: Use arules to discover patterns

Support is the proportion of times that a particular set of items occurs relative to the whole dataset.
Confidence is proportion of times that the consequent occurs when the antecedent is present.

A. Use **apriori** to generate a set of rules with support over 0.005 and confidence over 0.3, and trying to predict who successfully signed up for a term deposit.
Hint: You need to define the **right-hand side rule (rhs)**.

```
ruleset <- apriori(bankX, parameter=list(supp=0.005, conf=0.3),
control=list(verbose=F),appearance=list(default="lhs",rhs=("success=yes")))
```

B. Use inspect() to review of the **ruleset**.

```
inspect(ruleset)
```

```
##     lhs                                             rhs               support confidence   coverage
lift count
## [1] {job=student}                                => {success=yes} 0.006676702  0.3142857 0.02124405
2.789828   275
## [2] {job=student,
##      marital=single}                             => {success=yes} 0.006409634  0.3203883 0.02000583
2.843999   264
## [3] {job=student,
##      young=TRUE}                                 => {success=yes} 0.006579586  0.3180751 0.02068564
2.823465   271
## [4] {job=student,
##      contacted_before_this_campaign=FALSE} => {success=yes} 0.006676702  0.3142857 0.02124405
2.789828   275
## [5] {job=student,
##      marital=single,
##      young=TRUE}                                 => {success=yes} 0.006312518  0.3233831 0.01952025
2.870582   260
## [6] {job=student,
##      marital=single,
##      contacted_before_this_campaign=FALSE} => {success=yes} 0.006409634  0.3203883 0.02000583
2.843999   264
## [7] {job=student,
##      young=TRUE,
##      contacted_before_this_campaign=FALSE} => {success=yes} 0.006579586  0.3180751 0.02068564
2.823465   271
## [8] {job=student,
##      marital=single,
##      young=TRUE,
##      contacted_before_this_campaign=FALSE} => {success=yes} 0.006312518  0.3233831 0.01952025
2.870582   260
```

C. Use the output of inspect( ) or inspectDT( ) and describe **any 2 rules** the algorithm found.

```
#With 31.4% certainty, those with job = student will have successfully signed up for a term depo
sit.

#With 32% certainty, those with job = student and marital status = single will have successfully
signed up for a term deposit.
```

D. Generate a partition tree from the dataframe (not the transactions)

```
library(e1071)
library(rpart)
tree <- rpart(success~. , data=bank_key)
```

View the model (as a tree), and then explain with a block comment if this tree is helpful

```
library(rpart.plot)
rpart.plot(tree)
```


```
#No, the tree is not helpful, since this is an unsupervised model.
```