Analysis on Customer PEP Likelihood using Association Rule Mining

2024-01-29

Sarah Morris

Introduction and Business Question

The scope of this Data Analysis project is to determine which customers are most likely to obtain a PEP (Personal Equity Plan) based on current data we have on 600 customers with the bank. The Apriori Algorthim, which we will be using in this project, uses Association Rule Mining to find strong rule associations and possible predictors of whether a customer will be interested in opening a PEP. The goal of this project is to assist the business with optimal marketing resource allocation to target customers from certain demographics that will be most likely to obtain a PEP, based on current information we have about customers that already have opened a PEP with the bank.

Overview of the dataset

The bank data is in csv format and contains 600 observations of 11 attributes of customer data. The following table describes each attribute in depth and the original data type of each attribute.

Attribute	**Data Type**	**Description**
AGE	int	Age is a numeric value of each customer's age.
SEX	chr	Sex is a character with binary values of female or male.
REGION	chr	Region describes where the customer lives, and has four possible categories: inner city, rural, suburban, or town.
INCOME	num	Income is a numeric value of total dollar income per year.
MARRIED	chr	Married is a binary character that describes marital status as yes or no.

CHILDREN	int	Children is a number how many children (if any) the customer has.
CAR	chr	Car is a binary character that describes whether the customer owns a car as yes or no.
SAVE_ACT	chr	This is a binary character that describes whether the customer has a current saving account with the bank.
CURRENT_ACT	chr	This is a binary character that describes whether the customer has a current account with the bank.
MORTGAGE	chr	This is a binary character that describes whether the customer has a mortgage.
PEP	chr	This is a binary character that describes whether the customer currently has a PEP

Roadmap and Summary:

Roadmap:

1. Pre-processing steps
    a. Removal of ID field
    b. Convert numeric variables to discrete/nominal
2. Exploratory Data Analysis
    a. Check for missing values
    b. Simple visualizations of data
2. Association Rule Discovery
    a. Experiment with different parameters
    b. Identify 20-30 rules with high lift, confidence and support
3. Set PEP as RHS a. see which rules are generated

Summary:

1. Describe all above steps

a. Preprocessing

b. Parameters and experiments to find strong rules

2. Select 5 most interesting (not necessarily strongest) rules

a. Support, confidence, lift values

b. Explanation of patterns and why it is interesting based on company objectives

c. Any recommendations based on discovered rule to help company better understand customer behavior and develop business opportunities

3. For one rule, discuss support, confidence, and lift numbers and how they were computed from the data

Data Analysis

Read in Dataset:

```
bank <- read.csv("~/Downloads/week3_resources_2021/bankdata_csv_all.csv")
head(bank)

##       id age    sex     region  income married children car save_act
## 1 ID12101  48 FEMALE INNER_CITY 17546.0      NO        1  NO       NO
## 2 ID12102  40   MALE       TOWN 30085.1     YES        3 YES       NO
## 3 ID12103  51 FEMALE INNER_CITY 16575.4     YES        0 YES      YES
## 4 ID12104  23 FEMALE       TOWN 20375.4     YES        3  NO       NO
## 5 ID12105  57 FEMALE      RURAL 50576.3     YES        0  NO      YES
## 6 ID12106  57 FEMALE       TOWN 37869.6     YES        2  NO      YES
##   current_act mortgage pep
## 1          NO       NO YES
## 2         YES      YES  NO
## 3         YES       NO  NO
## 4         YES       NO  NO
## 5          NO       NO  NO
## 6         YES       NO YES
```

Preprocessing

To begin with data analysis, the ID field should be removed, as we do not need a primary key column for the association rule mining.

```
bank<-bank[,-1]
head(bank)

##   age    sex     region  income married children car save_act current_act
## 1  48 FEMALE INNER_CITY 17546.0      NO        1  NO       NO          NO
## 2  40   MALE       TOWN 30085.1     YES        3 YES       NO         YES
## 3  51 FEMALE INNER_CITY 16575.4     YES        0 YES      YES         YES
## 4  23 FEMALE       TOWN 20375.4     YES        3  NO       NO         YES
## 5  57 FEMALE      RURAL 50576.3     YES        0  NO      YES          NO
## 6  57 FEMALE       TOWN 37869.6     YES        2  NO      YES         YES
##   mortgage pep
## 1       NO YES
## 2      YES  NO
## 3       NO  NO
## 4       NO  NO
## 5       NO  NO
## 6       NO YES
```

Examining Structure of Dataset

```
str(bank)

## 'data.frame':    600 obs. of  11 variables:
##  $ age        : int  48 40 51 23 57 57 22 58 37 54 ...
##  $ sex        : chr  "FEMALE" "MALE" "FEMALE" "FEMALE" ...
##  $ region     : chr  "INNER_CITY" "TOWN" "INNER_CITY" "TOWN" ...
##  $ income     : num  17546 30085 16575 20375 50576 ...
##  $ married    : chr  "NO" "YES" "YES" "YES" ...
##  $ children   : int  1 3 0 3 0 2 0 0 2 2 ...
##  $ car        : chr  "NO" "YES" "YES" "NO" ...
##  $ save_act   : chr  "NO" "NO" "YES" "NO" ...
##  $ current_act: chr  "NO" "YES" "YES" "YES" ...
##  $ mortgage   : chr  "NO" "YES" "NO" "NO" ...
##  $ pep        : chr  "YES" "NO" "NO" "NO" ...
```

The dataset has 600 total rows or observations of customer data, and 11 attributes.

AGE: age is a numeric value of each customer's age (Integer).

SEX: sex is a character with binary values of female or male (Character).

REGION: region describes where the customer lives, and has four possible categories: inner city, rural, suburban, or town (Character).

INCOME: income is a numeric value of total dollar income per year (Number).

MARRIED: married is a binary character that describes marital status as yes or no (Character).

CAR: car is a binary character that describes whether the customer owns a car as yes or no (Character).

SAVE_ACT: this is a binary character that describes whether the customer has a current saving account with the bank (Character).

CURRENT_ACT: this is a binary character that describes whether the customer has a current account with the bank (Character).

MORTGAGE: this is a binary character that describes whether the customer has a mortgage (Character).

PEP: this is a binary character that describes whether the customer currently has a PEP (Character).

Discretizing, Binning, and Factorizing all Values

To proceed in our data analysis, we will convert all numeric values (age, income, and children) to discrete bins with the following values:

Convert age to bins: child, teenager, young adult, middle age, elderly

Convert income to bins: lower income (<$48,500), middle income ($48,501-$145,499), upper income (>$145,500)

Convert children to bins: No children (0) or Children (>1)

We will then convert all values to factors rather than characters.

```
bank$age<-cut(bank$age, breaks=c(0,10,20,40,65,Inf), labels=c("child", "teenager", "young adult", "middle-aged", "senior"))
bank$income<-cut(bank$income, breaks=c(0,48500,145500,Inf), labels=c("lower income", "middle income", "upper income"))
bank$children<-cut(bank$children, breaks=c(0,1,Inf), labels=c("No Children", "Children"))
```

```r
# view new structure of data
str(bank)
```

```
## 'data.frame':    600 obs. of  11 variables:
##  $ age        : Factor w/ 5 levels "child","teenager",..: 4 3 4 3 4 4 3 4 3 4 ...
##  $ sex        : chr  "FEMALE" "MALE" "FEMALE" "FEMALE" ...
##  $ region     : chr  "INNER_CITY" "TOWN" "INNER_CITY" "TOWN" ...
##  $ income     : Factor w/ 3 levels "lower income",..: 1 1 1 1 2 1 1 1 1 1 ...
##  $ married    : chr  "NO" "YES" "YES" "YES" ...
##  $ children   : Factor w/ 2 levels "No Children",..: 1 2 NA 2 NA 2 NA NA 2 2 ...
##  $ car        : chr  "NO" "YES" "YES" "NO" ...
##  $ save_act   : chr  "NO" "NO" "YES" "NO" ...
##  $ current_act: chr  "NO" "YES" "YES" "YES" ...
##  $ mortgage   : chr  "NO" "YES" "NO" "NO" ...
##  $ pep        : chr  "YES" "NO" "NO" "NO" ...
```

```r
#convert sex, region, married, car, save_act, current_act, mortgage, pep to factors
bank$sex<-factor(bank$sex)
bank$region<-factor(bank$region)
bank$married<-factor(bank$married)
bank$car<-factor(bank$car)
bank$save_act<-factor(bank$save_act)
bank$current_act<-factor(bank$current_act)
bank$mortgage<-factor(bank$mortgage)
bank$pep<-factor(bank$pep)
#examine structure of new df
str(bank)
```

```
## 'data.frame':    600 obs. of  11 variables:
##  $ age        : Factor w/ 5 levels "child","teenager",..: 4 3 4 3 4 4 3 4 3 4 ...
##  $ sex        : Factor w/ 2 levels "FEMALE","MALE": 1 2 1 1 1 1 2 2 1 2 ...
##  $ region     : Factor w/ 4 levels "INNER_CITY","RURAL",..: 1 4 1 4 2 4 2 4 3 4 ...
##  $ income     : Factor w/ 3 levels "lower income",..: 1 1 1 1 2 1 1 1 1 1 ...
##  $ married    : Factor w/ 2 levels "NO","YES": 1 2 2 2 2 2 1 2 2 2 ...
##  $ children   : Factor w/ 2 levels "No Children",..: 1 2 NA 2 NA 2 NA NA 2 2 ...
```

```
##  $ car        : Factor w/ 2 levels "NO","YES": 1 2 2 1 1 1 1 2 2 2 ...
##  $ save_act   : Factor w/ 2 levels "NO","YES": 1 1 2 1 2 2 1 2 1 2 ...
##  $ current_act: Factor w/ 2 levels "NO","YES": 1 2 2 2 1 2 2 2 1 2 ...
##  $ mortgage   : Factor w/ 2 levels "NO","YES": 1 2 1 1 1 1 1 1 1 1 ...
##  $ pep        : Factor w/ 2 levels "NO","YES": 2 1 1 1 1 2 2 1 1 1 ...
```

Exploratory Data Analysis

The data is now preprocessed, so we will proceed with exploratory data analysis.

Before using the Apriori Algorithm for association rule mining, it is important to get a better feel for the data and the relative frequencies of each category of customers. This is useful because it provides context to the rules that will be developed. If the majority of customers fall into a certain category for various attributes, this will be reflected in the rule sets as well. This context will be necessary to understand if rules are actually strong, or simply common based on the customer demographics.

Checking for NA values in attributes

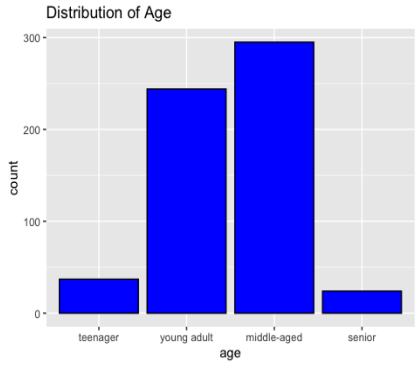
Before proceeding with data analysis, we will check for missing values.

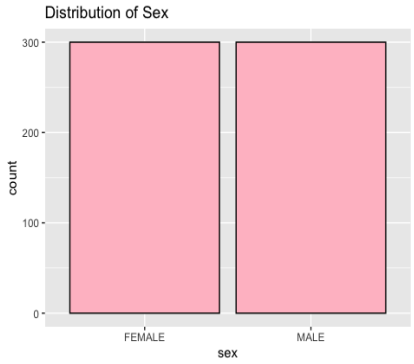
```
#checking for missing values
sum(is.na(bank$age))

## [1] 0

#no missing values for age
sum(is.na(bank$sex))

## [1] 0

#no missing values for sex
sum(is.na(bank$region))

## [1] 0

#no missing values for region
sum(is.na(bank$income))

## [1] 0
```

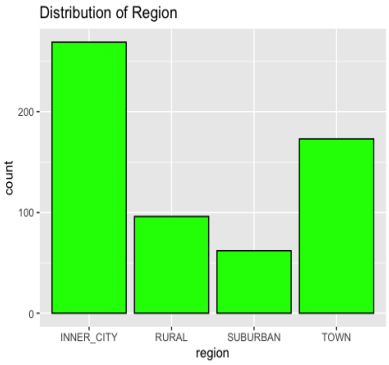
```
#no missing values for income
sum(is.na(bank$married))

## [1] 0

#no missing values for married
sum(is.na(bank$children))

## [1] 263

#263 missing values for children
sum(is.na(bank$car))

## [1] 0

#no missing values for car
sum(is.na(bank$save_act))

## [1] 0

#no missing values for save_act
sum(is.na(bank$current_act))

## [1] 0

#no missing values for current_act
sum(is.na(bank$mortgage))

## [1] 0

#no missing values for mortgage
sum(is.na(bank$pep))

## [1] 0

#no missing values for PEP
```

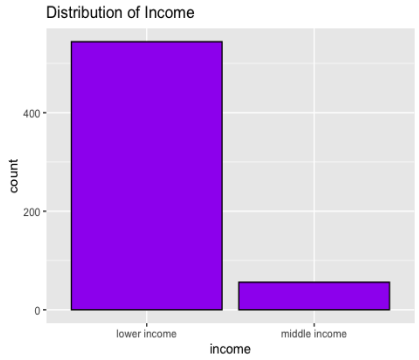
There are only missing values for one attribute: whether the customer has children. It does not make sense to use mean here, since we are working with a binary factor {Children} or {No Children}. Therefore, we will simply note that there are unknown values in this column while we proceed with our data analysis.

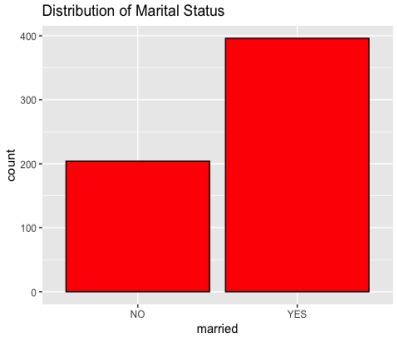
Simple Visualizations of Dataset using GGPLOT

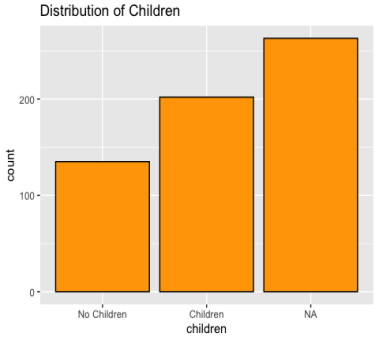


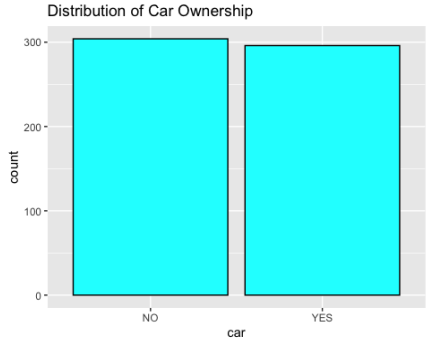


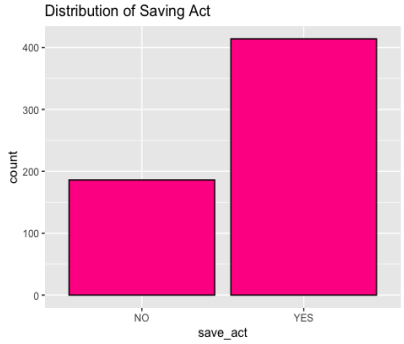


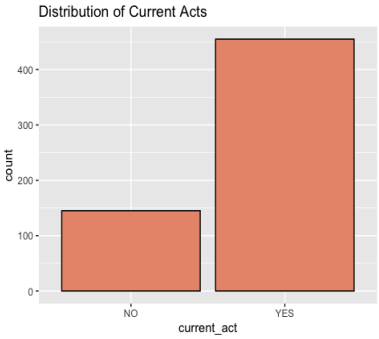


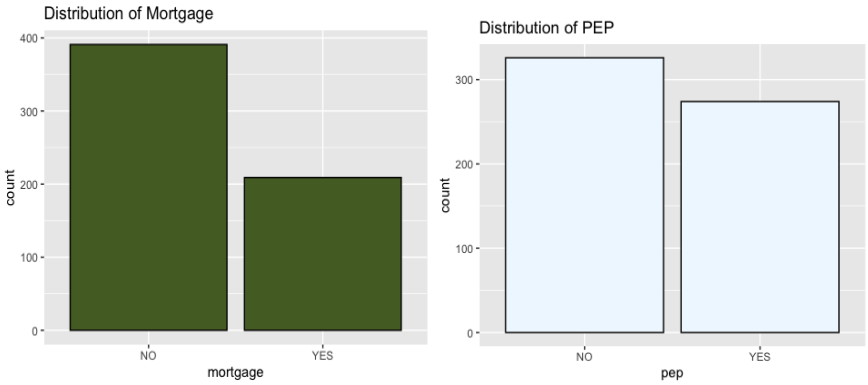












Summary of percentages of the most frequent items:

```
(sum(bank$age=="middle-aged")/length(bank$age))*100
```

```
## [1] 49.16667
```

```
(sum(bank$age=="young adult")/length(bank$age))*100
```

```
## [1] 40.66667
```

```
(sum(bank$sex=="FEMALE")/length(bank$sex))*100
```

```
## [1] 50
```

```
(sum(bank$region=="INNER_CITY")/length(bank$region))*100
```

```
## [1] 44.83333
```

```
(sum(bank$income=="lower income")/length(bank$income))*100
```

```
## [1] 90.66667
```

```
(sum(bank$married=="YES")/length(bank$married))*100
```

```
## [1] 66
```

```
(sum(bank$car=="NO")/length(bank$car))*100
```

```
## [1] 50.66667
```

```
(sum(bank$save_act=="YES")/length(bank$save_act))*100
```

```
## [1] 69
```

```
(sum(bank$current_act=="YES")/length(bank$current_act))*100
```

```
## [1] 75.83333
```

```
(sum(bank$mortgage=="NO")/length(bank$mortgage))*100
```

```
## [1] 65.16667
```

```
(sum(bank$pep=="NO")/length(bank$pep))*100
```

```
## [1] 54.33333
```

AGE:49% of customers are middle-aged. 41% are young adults.

SEX: 50% of customers are female, 50% are male.

REGION: 45% of customers live in the inner city.

INCOME: 91% of customers are lower income.

MARITAL STATUS: 66% of customers are married.

CHILDREN: The majority the children attribute is unknown.

CAR: 51% of customers do not own a car.

SAVING ACCOUNT: 69% of customers have a saving account.

CURRENT ACCOUNT: 76% of customers have a current account.

MORTGAGE: 65% of customers do not have a mortgage.

PEP: 54% of customers do not have a PEP.

Association Rule Discovery

Based on the above analysis, there is now more context to the dataset. We can now proceed with using the Apriori algorithm to determine most frequent rules in the dataset.

First, read in Apriori Algorithm libraries:

```
library(arules)
```

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

```
##
## Attaching package: 'Matrix'
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack

##
## Attaching package: 'arules'

## The following object is masked from 'package:dplyr':
##
##     recode

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

library(arulesViz)
```

Summarizing Ranges and Means of Confidence, Support, and Lift for Rules

First we will summarize the rules that are at least a 10% frequency (support) and 80% confidence.

```
rules<-apriori(bank, parameter = list(sup=0.1, conf=0.8))

## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.8    0.1    1 none FALSE            TRUE       5     0.1      1
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 60
##
```

```
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[26 item(s), 600 transaction(s)] done [0.00s].
## sorting and recoding items ... [23 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 5 6 done [0.00s].
## writing ... [731 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
summary(rules)
```

```
## set of 731 rules
##
## rule length distribution (lhs + rhs):sizes
##   1   2   3   4   5   6
##   1  22 175 343 166  24
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   3.000   4.000   3.989   5.000   6.000
##
## summary of quality measures:
##     support         confidence        coverage           lift
##  Min.   :0.1000   Min.   :0.8000   Min.   :0.1000   Min.   :0.8824
##  1st Qu.:0.1117   1st Qu.:0.8472   1st Qu.:0.1250   1st Qu.:0.9729
##  Median :0.1333   Median :0.8989   Median :0.1517   Median :1.0284
##  Mean   :0.1602   Mean   :0.9020   Mean   :0.1781   Mean   :1.0453
##  3rd Qu.:0.1792   3rd Qu.:0.9533   3rd Qu.:0.1933   3rd Qu.:1.1029
##  Max.   :0.9067   Max.   :1.0000   Max.   :1.0000   Max.   :2.0959
##      count
##  Min.   : 60.00
##  1st Qu.: 67.00
##  Median : 80.00
##  Mean   : 96.09
##  3rd Qu.:107.50
##  Max.   :544.00
```

```
##
## mining info:
##  data ntransactions support confidence
##  bank          600     0.1        0.8
##                                           call
##  apriori(data = bank, parameter = list(sup = 0.1, conf = 0.8))
```

From this command, we find that we have 731 rules that meet our criteria. Our max support level is .91. Our max confidence level is 1, and our max lift level is 2.10. This will give us a good idea of which rules are the strongest as we proceed with our analysis.

High Confidence Rules

First we will look for rules with a support of at least 0.1 (occurs at a frequency of at least 10% of the time) and a confidence level of at least 0.8 (rounding the decimals to 2 places). We will take a look at the first 50 rules.[1]

```
rules_conf<-apriori(bank, parameter = list(sup=0.1, conf=0.8))

## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.8    0.1    1 none FALSE            TRUE       5     0.1      1
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 60
##
## set item appearances ...[0 item(s)] done [0.00s].
```

---

[1] While I analyzed 50 rules at a time in R, I am only including 5 on this document for brevity. The 20-30 extracted rules can be seen in the next section.

```
## set transactions ...[26 item(s), 600 transaction(s)] done [0.00s].
## sorting and recoding items ... [23 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 5 6 done [0.00s].
## writing ... [731 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
rules_conf<-sort(rules_conf, by="confidence", decreasing=TRUE)
options(digits=2)
inspect(rules_conf[1:50])
```

```
##     lhs                    rhs                  support confidence coverage lift count
## [1] {save_act=NO}       => {income=lower income}    0.31          1     0.31  1.1    186
## [2] {age=young adult}   => {income=lower income}    0.41          1     0.41  1.1    244
## [3] {age=young adult,
##      current_act=NO}    => {income=lower income}    0.11          1     0.11  1.1     65
## [4] {age=young adult,
##      region=TOWN}       => {income=lower income}    0.13          1     0.13  1.1     77
## [5] {married=NO,
##      save_act=NO}       => {income=lower income}    0.11          1     0.11  1.1     67
```

All rules 1-100 have a lift of 1.1, which is relatively weak. These rules also show us that the customers are likely to have a low income (which we know from EDA is 91% likely given our dataset). Therefore, these rules are not useful for our purposes.

High Support Rules

Now, filtering for rules with support greater than .4, and a confidence of only .1. This will surface the rules with a stronger support (frequency of occurrence).

```
rules_sup<-apriori(bank, parameter = list(sup=0.3, conf=0.1))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.1    0.1    1 none FALSE            TRUE       5     0.3      1
```

```
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 180
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[26 item(s), 600 transaction(s)] done [0.00s].
## sorting and recoding items ... [18 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 done [0.00s].
## writing ... [202 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
rules_sup<-sort(rules_sup, by="support", decreasing=TRUE)
options(digits=2)
inspect(rules_sup[1:30])
```

```
##      lhs                    rhs                  support confidence coverage
## [1]  {}                  => {income=lower income} 0.91    0.91       1.00
## [2]  {}                  => {current_act=YES}     0.76    0.76       1.00
## [3]  {}                  => {save_act=YES}        0.69    0.69       1.00
## [4]  {current_act=YES}     => {income=lower income} 0.68    0.90       0.76
## [5]  {income=lower income} => {current_act=YES}     0.68    0.75       0.91

##      lift count
## [1]  1.00 544
## [2]  1.00 455
## [3]  1.00 414
## [4]  0.99 410
## [5]  0.99 410
```

The rules with the highest support often have a missing left hand side, which is not useful for our purposes. It will be more useful to filter by lift (finding the strongest rules in the dataset.) We will start with parameters of 0.1 support and 0.8 confidence.

High Lift Rules

Sorting these rules for highest lift:

```
rules<-apriori(bank, parameter = list(sup=0.1, conf=0.8))

## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.8    0.1    1 none FALSE            TRUE       5     0.1      1
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 60
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[26 item(s), 600 transaction(s)] done [0.00s].
## sorting and recoding items ... [23 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 5 6 done [0.00s].
## writing ... [731 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].

rules<-sort(rules, by="lift", decreasing=TRUE)
options(digits=2)
inspect(rules[1:30])
```

```
##      lhs                     rhs          support confidence coverage lift count
## [1]  {age=middle-aged,
##       children=No Children}  => {pep=YES}       0.11       0.96     0.12  2.1    67
## [2]  {children=No Children,
##       save_act=YES,
##       current_act=YES}       => {pep=YES}       0.10       0.86     0.12  1.9    63
## [3]  {children=No Children,
##       mortgage=NO}           => {pep=YES}       0.12       0.85     0.14  1.9    71
## [4]  {children=No Children,
##       save_act=YES}          => {pep=YES}       0.13       0.84     0.16  1.8    80
## [5]  {children=No Children,
##       current_act=YES}       => {pep=YES}       0.14       0.83     0.17  1.8    84
```

The rules with the highest lift are those with PEP on the RHS, which will be covered later in our data analysis.

Exploring Combinations and identifying 20-30 interesting rules

Now sorting for rules with confidence and support over 0.5:

```
rules_sup_conf<-apriori(bank, parameter = list(sup=0.5, conf=0.5))

## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.5    0.1    1 none FALSE            TRUE       5     0.5      1
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 300
```

```
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[26 item(s), 600 transaction(s)] done [0.00s].
## sorting and recoding items ... [9 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 done [0.00s].
## writing ... [23 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
options(digits=2)
inspect(rules_sup_conf[1:20])
```

```
##      lhs                 rhs                 support confidence coverage
## [1]  {}               => {sex=FEMALE}         0.50    0.50       1.00
## [2]  {}               => {sex=MALE}           0.50    0.50       1.00
## [3]  {}               => {car=NO}             0.51    0.51       1.00
## [4]  {}               => {pep=NO}             0.54    0.54       1.00
## [5]  {}               => {mortgage=NO}        0.65    0.65       1.00

##      lift count
## [1]  1.00 300
## [2]  1.00 300
## [3]  1.00 304
## [4]  1.00 326
## [5]  1.00 391
```

For this exercise we will ignore any rules with a blank LHS. The 11 strongest rules for support, confidence, and lift (measures are in order, last metric is "count"):

[10] {pep=NO} => {income=lower income} 0.52 0.96 1.06 313

[11] {income=lower income} => {pep=NO} 0.52 0.58 1.06 313

[12] {mortgage=NO} => {current_act=YES} 0.50 0.77 1.02 301

[13] {current_act=YES} => {mortgage=NO} 0.50 0.66 1.02 301

[14] {mortgage=NO} => {income=lower income} 0.59 0.91 1.00 354

[15] {income=lower income} => {mortgage=NO} 0.59 0.65 1.00 354

[16] {married=YES} => {income=lower income} 0.60 0.91 1.00 360

[17] {income=lower income} => {married=YES} 0.60 0.66 1.00 360

[18] {save_act=YES} => {current_act=YES} 0.53 0.77 1.02 319

[19] {current_act=YES} => {save_act=YES} 0.53 0.70 1.02 319 [20] {save_act=YES} => {income=lower income} 0.60 0.86 0.95 358

Now for rules with a minimum support of .3 and a confidence of at least .9

```
rules_strong_conf<-apriori(bank, parameter = list(sup=0.3, conf=0.9))

## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.9    0.1    1 none FALSE            TRUE       5     0.3      1
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 180
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[26 item(s), 600 transaction(s)] done [0.00s].
## sorting and recoding items ... [18 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 done [0.00s].
## writing ... [26 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].

options(digits=2)
inspect(rules_strong_conf[1:20])
```

```
##      lhs                    rhs                    support confidence coverage lift count
## [1]  {}                  => {income=lower income}    0.91       0.91     1.00 1.00   544
## [2]  {save_act=NO}       => {income=lower income}    0.31       1.00     0.31 1.10   186
## [3]  {married=NO}        => {income=lower income}    0.31       0.90     0.34 0.99   184
## [4]  {mortgage=YES}      => {income=lower income}    0.32       0.91     0.35 1.00   190
## [5]  {age=young adult}   => {income=lower income}    0.41       1.00     0.41 1.10   244
```

Strongest rules with a minimum threshold of .9 confidence and .3 support (stronger confidence compared to support in comparison to previous rules):

[2] {save_act=NO} => {income=lower income} 0.31 1.00 1.10 186

[3] {married=NO} => {income=lower income} 0.31 0.90 0.99 184

[4] {mortgage=YES} => {income=lower income} 0.32 0.91 1.00 190

[5] {age=young adult} => {income=lower income} 0.41 1.00 1.10 244

[6] {region=INNER_CITY} => {income=lower income} 0.41 0.92 1.01 247

[7] {sex=FEMALE} => {income=lower income} 0.45 0.90 1.00 271

[8] {sex=MALE} => {income=lower income} 0.46 0.91 1.00 273

[9] {car=NO} => {income=lower income} 0.47 0.92 1.01 279

[11] {mortgage=NO} => {income=lower income} 0.59 0.91 1.00 354

[12] {married=YES} => {income=lower income} 0.60 0.91 1.00 360

[13] {current_act=YES} => {income=lower income} 0.68 0.90 0.99 410

[14] {region=INNER_CITY, current_act=YES} => {income=lower income} 0.31 0.90 1.00 185

[15] {sex=MALE, married=YES} => {income=lower income} 0.30 0.90 0.99 181

[16] {sex=MALE, current_act=YES} => {income=lower income} 0.34 0.92 1.01 206

[17] {car=NO, mortgage=NO} => {income=lower income} 0.30 0.92 1.01 181

[18] {married=YES, car=NO} => {income=lower income} 0.31 0.93 1.02 187

[19] {car=NO, current_act=YES} => {income=lower income} 0.36 0.91 1.01 215

[20] {mortgage=NO, pep=NO} => {income=lower income} 0.34 0.97 1.07 202

Note that some of these rules are in opposition to one another: For example, {save_act=NO} => {income=lower income} is in opposition to {save_act=YES} => {income=lower income}. The rules {sex=FEMALE} => {income=lower income} and

{sex=MALE} => {income=lower income} are also not very helpful for our purposes. Based on our previous context, we know that there are 50% Females and 50% Males in the Dataset, and 91% of all candidates are low income. Therefore, this rule is pretty self explanatory and doesn't illustrate anything new to us.

Placing PEP on RHS

After previously sorting by lift, we saw that the strongest rules were those with PEP on the right hand side. Now, we will filter rules by placing PEP on the RHS.

```
rules<-apriori(data=bank, parameter=list(supp=0.1,conf=0.3,minlen=2),
               appearance=list(default="lhs",rhs="pep=YES"),
               control = list(verbose=F))
rules<-sort(rules, decreasing=TRUE,by="confidence")
inspect(rules[1:30])

##     lhs                        rhs       support confidence coverage lift count
## [1] {age=middle-aged,
##      children=No Children}  => {pep=YES}    0.11       0.96     0.12  2.1    67
## [2] {children=No Children,
##      save_act=YES,
##      current_act=YES}       => {pep=YES}    0.10       0.86     0.12  1.9    63
## [3] {children=No Children,
##      mortgage=NO}           => {pep=YES}    0.12       0.85     0.14  1.9    71
## [4] {children=No Children,
##      save_act=YES}          => {pep=YES}    0.13       0.84     0.16  1.8    80
## [5] {children=No Children,
##      current_act=YES}       => {pep=YES}    0.14       0.83     0.17  1.8    84
```

Examining 5 most interesting rules from PEP on RHS

There are several interesting rules that come from this analysis, and these rules are more illustrative than the previous rules that were found above. The following 5 rules are especially important in determining whether a customer will also have a PEP:

1. {age=middle-aged, children=No Children} => {pep=YES} Support: .11 –> 11% of all customers are middle aged, have no children, and have a PEP. Confidence: .96 –> If a customer is middle aged and has no children, they are 96% likely to have a PEP. Lift:

2.1 –> A customer with a PEP is 2.1 times likely to be middle aged and have no children.

This rule illustrates that middle aged people with no children are statistically most likely to also have a PEP. However, based on our previous context, we know that many people did not report whether they have children or not. This is a gap in the data that should be kept in mind when analyzing this rule. If we had more data about who has children and who does not, it would likely change the strength of this rule. This rule does represent a business opportunity - the bank should consider reaching out to more middle aged people to offer them a PEP, especially if they don't have kids.

2. {children=No Children, save_act=YES, current_act=YES} => {pep=YES} Support: 0.10 Confidence: 0.86 Lift: 1.9

This rule tells us that customers with a savings account and current account (who also do not have children) are 86% likely to also have a PEP. (Again, the children factor should be taken with a grain of salt due to missing values.) This represents a business opportunity for the bank to reach out to people with two open accounts to also set up a PEP.

3. {children=No Children, mortgage=NO} => {pep=YES} Support: 0.12 Confidence: 0.85 Lift: 1.9

This rule is interesting because it is counter intuitive - stating that customers WITHOUT a mortgage (and no children) are more likely (85%) to have a PEP. We see again that people with no children are likely to have a PEP. Despite our previous notes, this seems to be a strong rule and would be worth investigating further. The bank should consider reaching out to people with no mortgage and no children to see if they would be interested to set up a PEP.

4. {income=lower income, children=No Children, mortgage=NO} => {pep=YES} Support: .1 Confidence: 0.83 Lift: 1.8

This rule reflects what we know from the previous rules, stating that people with lower income, no children and no mortgage are 83% likely to also have a PEP. Considering that 91% of people in the dataset are low income, it is likely that lower income is included in this rule simply due to the high frequency of lower income customers in the dataset. This rule echoes the business strategy of targeting customers with no morgage and no children to open a PEP.

5. {income=lower income, children=No Children, save_act=YES} => {pep=YES} Support: .11 Confidence: .82 Lift: 1.8

This rule also echoes the previous factors that we know to be influential in whether a customer has a PEP: if they have current accounts open, if they don't have children, and if they are low income. The bank should consider reaching out to people with current saving accounts who also don't have children.

Summary

To recap the analysis, we first pre-processed the data to discretize numeric values into "bins" and checked for missing values in the data that would represent any knowledge gaps. It was evident that the "children" attribute was missing for nearly half of all customers. This is a large knowledge gap that is important to factor into our analysis, especially because the children factor was strongly associated with whether customers have a PEP.

We then looked at each individual attribute to see the proportion of customers in each bin. Age, region, income, and current accounts had the largest spread among all of our attributes. Income was the most heavily skewed, with 91% of all customers being lower-income and no customers being upper income. This was important knowledge for us to consider when analyzing our rules, as lower income appeared often due to the high proportion of customers in this category.

When using the Apriori algorithm, we looked at rules with high confidence, then rules with high support, and finally rules with high lift. Rules with high confidence had a relatively low lift (1.1 in most cases), and had a "low income" on the RHS. Since 91% of the dataset is low income, this was not helpful for data analysis. Rules with a high support were more likely to have an empty left hand side, which was also not useful for our data analysis. Finally, rules with the highest lift were the most illustrative, and were more likely to have {PEP=YES} on the RHS.

Finally, we manually placed {PEP=YES} to examine these rules. These rules were the most important for our analysis and for developing business insights. We identified 5 important rules that had a lift of 1.8 or over (strong rules), a confidence of at least 82%, and a support (frequency) of at least 10% in our dataset.

Conclusion and Business Application

The rules derived from placing {PEP=YES} on the RHS showed that the following factors are MOST important to determine whether a customer will have a PEP (in no particular order): 1. Middle-Aged 2. Savings Account Open 3. Current Account Open 4. No Mortgage 5. No Children *6. Lower Income*

The final two factors were slightly problematic in our rule analysis. Lower Income represented 91% of all customers, so it is difficult to tell whether Lower Income is a predictor of a PEP or whether it is simply overrepresented in the dataset. Regardless, it surfaced in 2/5 of the strongest rules, so it is worth considering as a factor.

Finally, the most problematic attribute was whether the customers have children or not. In 5/5 of the strongest rules, {No Children} was a predictor for whether a customer has a PEP. However, the dataset has nearly 50% of the values unknown for this attribute - that is, we do not know whether 50% of our customers have children or not. My recommendation is for the bank to gather more data regarding this attribute to better understand the importance of No Children for a customer having a PEP.

However, based on this analysis, the bank now knows that Middle Aged Customers that do not have a mortgage and already have savings and current accounts open with the bank are excellent candidates for PEPs. The Bank should primarily target the customers that do not have PEPs are already satisfy the previous requirements, as they will be most likely to open a PEP based on this analysis.