HW3\_IST707

Hana Kim

10/26/2019

## Introduction

Customer data is information held on file about customers by a store or other business, usually including names, contact details, and buying habits. Customer data are the firsthand responses that are obtained from customers through investigation or by asking direct questions. (<https://www.collindictionary.com>, customer data definition and meaning) By analyzing customer data, business can obtain valuable information about customers behavior and patterns that can help shape and direct the business’ plans and activities.

## Analysis and Model

About the Data

The marketing department of a financial firm keeps records on customers, including demographic information and number of type of accounts. When launching a new product, such as a “Personal Equity Plan” (PEP), a direct mail piece, advertising the product is sent to existing customer and a record kept as to whether that customer responded and bought the product.

Based on the prior experience in the dataset, a data mining technique is used to build customer profile models.

The dataset, bankdata.csv, contains the following fields:

id, age, sex, region, income, married, children, car, save\_acct, current\_acct, mortgage, pep (a customer buy a PEP after the last mailing)

The marketing department of a financial firm keeps records on customers, including demographic information and number of type of accounts. When launching a new product, such as a “Personal Equity Plan” (PEP), a direct mail piece, advertising the product is sent to existing customer and a record kept as to whether that customer responded and bought

the product.

When the dataset is loaded, it is shown as below:

bd=read.csv('/Users/hana/Documents/IST 707 Data Analytics/bankdata.csv')  
str(bd)

## 'data.frame': 600 obs. of 12 variables:  
## $ id : Factor w/ 600 levels "ID12101","ID12102",..: 1 2 3 4 5 6 7 8 9 10 ...  
## $ age : int 48 40 51 23 57 57 22 58 37 54 ...  
## $ 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 : num 17546 30085 16575 20375 50576 ...  
## $ married : Factor w/ 2 levels "NO","YES": 1 2 2 2 2 2 1 2 2 2 ...  
## $ children : int 1 3 0 3 0 2 0 0 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 ...

In order to use a data mining technique, preprocessing of the dataset is required as variables are in different data types. One way of discretization and numeric-to-nominal conversion is using different bins.

discretization and numeric-to-nominal transformation

bd$age <- cut(bd$age, breaks = c(0,10,20,30,40,50,60,Inf),labels=c("child","teens","twenties","thirties","fourties","fifties","old"))

discretize income by equal-width bin

min\_income <- min(bd$income)  
max\_income <- max(bd$income)  
bins = 3   
width=(max\_income - min\_income)/bins;  
bd$income = cut(bd$income, breaks=seq(min\_income, max\_income, width))

convert numeric to nominal for ‘children’ variable

bd$children=factor(bd$children)

convert “YES” “NO” data type to ‘[variable\_name]=YES’’[variable\_name]=NO’

bd$married=dplyr::recode(bd$married, YES="married=YES", NO="married=NO")  
bd$car=dplyr::recode(bd$car, YES="car=YES", NO="car=NO")  
bd$save\_act=dplyr::recode(bd$save\_act, YES="save\_act=YES", NO="save\_act=NO")  
bd$current\_act=dplyr::recode(bd$current\_act, YES="current\_act=YES", NO="current\_act=NO")  
bd$mortgage=dplyr::recode(bd$mortgage, YES="mortgage=YES", NO="mortgage=NO")  
bd$pep=dplyr::recode(bd$pep, YES="pep=YES", NO="pep=NO")

After the preprocessing of the dataset, all variables have the same type of data type, Factor

str(bd)

## 'data.frame': 600 obs. of 12 variables:  
## $ id : Factor w/ 600 levels "ID12101","ID12102",..: 1 2 3 4 5 6 7 8 9 10 ...  
## $ age : Factor w/ 7 levels "child","teens",..: 5 4 6 3 6 6 3 6 4 6 ...  
## $ 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 "(5.01e+03,2.44e+04]",..: 1 2 1 1 3 2 1 2 2 1 ...  
## $ married : Factor w/ 2 levels "married=NO","married=YES": 1 2 2 2 2 2 1 2 2 2 ...  
## $ children : Factor w/ 4 levels "0","1","2","3": 2 4 1 4 1 3 1 1 3 3 ...  
## $ car : Factor w/ 2 levels "car=NO","car=YES": 1 2 2 1 1 1 1 2 2 2 ...  
## $ save\_act : Factor w/ 2 levels "save\_act=NO",..: 1 1 2 1 2 2 1 2 1 2 ...  
## $ current\_act: Factor w/ 2 levels "current\_act=NO",..: 1 2 2 2 1 2 2 2 1 2 ...  
## $ mortgage : Factor w/ 2 levels "mortgage=NO",..: 1 2 1 1 1 1 1 1 1 1 ...  
## $ pep : Factor w/ 2 levels "pep=NO","pep=YES": 2 1 1 1 1 2 2 1 1 1 ...

library(magrittr)  
library(plyr)  
library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:plyr':  
##   
## arrange, count, desc, failwith, id, mutate, rename, summarise,  
## summarize

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

library(arules)

## Loading required package: Matrix

##   
## Attaching package: 'arules'

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

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

## Then, load the dataset

bd=read.csv('/Users/hana/Documents/IST 707 Data Analytics/bankdata.csv')  
str(bd)

## 'data.frame': 600 obs. of 12 variables:  
## $ id : Factor w/ 600 levels "ID12101","ID12102",..: 1 2 3 4 5 6 7 8 9 10 ...  
## $ age : int 48 40 51 23 57 57 22 58 37 54 ...  
## $ 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 : num 17546 30085 16575 20375 50576 ...  
## $ married : Factor w/ 2 levels "NO","YES": 1 2 2 2 2 2 1 2 2 2 ...  
## $ children : int 1 3 0 3 0 2 0 0 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 ...

## First necessary preprocessing step of conversion:

## discretization and numeric-to-nominal transformation

Preprocessing of the dataset is required as variables are in different datat types. One way of descretization and numeric-to-nominal conversion is using different bins. ##Discretize age by age group bin

bd$age <- cut(bd$age, breaks = c(0,10,20,30,40,50,60,Inf),labels=c("child","teens","twenties","thirties","fourties","fifties","old"))

## Discretize income by equal-width bin

min\_income <- min(bd$income)  
max\_income <- max(bd$income)  
bins = 3   
width=(max\_income - min\_income)/bins;  
bd$income = cut(bd$income, breaks=seq(min\_income, max\_income, width))

## Convert numeric to nominal for ‘children’ variable

bd$children=factor(bd$children)

## Convert “YES”“NO” data type to “[variable\_name]=YES”“[variable\_name]=NO”

bd$married=dplyr::recode(bd$married, YES="married=YES", NO="married=NO")  
bd$car=dplyr::recode(bd$car, YES="car=YES", NO="car=NO")  
bd$save\_act=dplyr::recode(bd$save\_act, YES="save\_act=YES", NO="save\_act=NO")  
bd$current\_act=dplyr::recode(bd$current\_act, YES="current\_act=YES", NO="current\_act=NO")  
bd$mortgage=dplyr::recode(bd$mortgage, YES="mortgage=YES", NO="mortgage=NO")  
bd$pep=dplyr::recode(bd$pep, YES="pep=YES", NO="pep=NO")

## After the preprocessing of dataset, all variables have the same type of data type, Factor

str(bd)

## 'data.frame': 600 obs. of 12 variables:  
## $ id : Factor w/ 600 levels "ID12101","ID12102",..: 1 2 3 4 5 6 7 8 9 10 ...  
## $ age : Factor w/ 7 levels "child","teens",..: 5 4 6 3 6 6 3 6 4 6 ...  
## $ 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 "(5.01e+03,2.44e+04]",..: 1 2 1 1 3 2 1 2 2 1 ...  
## $ married : Factor w/ 2 levels "married=NO","married=YES": 1 2 2 2 2 2 1 2 2 2 ...  
## $ children : Factor w/ 4 levels "0","1","2","3": 2 4 1 4 1 3 1 1 3 3 ...  
## $ car : Factor w/ 2 levels "car=NO","car=YES": 1 2 2 1 1 1 1 2 2 2 ...  
## $ save\_act : Factor w/ 2 levels "save\_act=NO",..: 1 1 2 1 2 2 1 2 1 2 ...  
## $ current\_act: Factor w/ 2 levels "current\_act=NO",..: 1 2 2 2 1 2 2 2 1 2 ...  
## $ mortgage : Factor w/ 2 levels "mortgage=NO",..: 1 2 1 1 1 1 1 1 1 1 ...  
## $ pep : Factor w/ 2 levels "pep=NO","pep=YES": 2 1 1 1 1 2 2 1 1 1 ...

Model

The model used to analyze the customer profile is Apriori algorithm for finding frequent item sets in a dataset.

## The apriori algorithm is run with the transformed data

## supp=0.001,conf=0.9,maxlen=3

rules = apriori(bd, parameter = list(supp = 0.001, conf = 0.9, maxlen = 3))

## Apriori  
##   
## Parameter specification:  
## confidence minval smax arem aval originalSupport maxtime support minlen  
## 0.9 0.1 1 none FALSE TRUE 5 0.001 1  
## maxlen target ext  
## 3 rules FALSE  
##   
## Algorithmic control:  
## filter tree heap memopt load sort verbose  
## 0.1 TRUE TRUE FALSE TRUE 2 TRUE  
##   
## Absolute minimum support count: 0   
##   
## set item appearances ...[0 item(s)] done [0.00s].  
## set transactions ...[631 item(s), 600 transaction(s)] done [0.00s].  
## sorting and recoding items ... [631 item(s)] done [0.00s].  
## creating transaction tree ... done [0.00s].  
## checking subsets of size 1 2 3

## Warning in apriori(bd, parameter = list(supp = 0.001, conf = 0.9, maxlen  
## = 3)): Mining stopped (maxlen reached). Only patterns up to a length of 3  
## returned!

## done [0.00s].  
## writing ... [72691 rule(s)] done [0.01s].  
## creating S4 object ... done [0.01s].

options(digits=2)  
inspect(rules[1:5])

## lhs rhs support confidence lift count  
## [1] {id=ID12165} => {age=twenties} 0.0017 1 5.0 1   
## [2] {id=ID12165} => {region=TOWN} 0.0017 1 3.5 1   
## [3] {id=ID12165} => {married=married=NO} 0.0017 1 2.9 1   
## [4] {id=ID12165} => {mortgage=mortgage=YES} 0.0017 1 2.9 1   
## [5] {id=ID12165} => {children=0} 0.0017 1 2.3 1

## First look at the apriori algorithm output

As shwon above, the outcome of the rules does not provide any meaningful information  
A reason is that the ‘id’ variable is associating with other variable with full confidence which is obvious, because other variabls are pertained to each ‘id’ variables.  
So second preporcessing is required to remove ‘id’ column.  
## Drop the id variable, convert categorical variable to factor, discretize numeric variables

bd <- bd %>%   
 select(-id) %>%   
 mutate\_if(is.character, funs(as.factor)) %>%   
 mutate\_if(is.numeric, funs(discretize))

## Test run with the algorithm

rules = apriori(bd, parameter = list(supp = 0.001, conf = 0.9, maxlen = 3))

## Apriori  
##   
## Parameter specification:  
## confidence minval smax arem aval originalSupport maxtime support minlen  
## 0.9 0.1 1 none FALSE TRUE 5 0.001 1  
## maxlen target ext  
## 3 rules FALSE  
##   
## Algorithmic control:  
## filter tree heap memopt load sort verbose  
## 0.1 TRUE TRUE FALSE TRUE 2 TRUE  
##   
## Absolute minimum support count: 0   
##   
## set item appearances ...[0 item(s)] done [0.00s].  
## set transactions ...[31 item(s), 600 transaction(s)] done [0.00s].  
## sorting and recoding items ... [31 item(s)] done [0.00s].  
## creating transaction tree ... done [0.00s].  
## checking subsets of size 1 2 3

## Warning in apriori(bd, parameter = list(supp = 0.001, conf = 0.9, maxlen  
## = 3)): Mining stopped (maxlen reached). Only patterns up to a length of 3  
## returned!

## done [0.00s].  
## writing ... [112 rule(s)] done [0.00s].  
## creating S4 object ... done [0.00s].

options(digits=2)  
inspect(rules[1:5])

## lhs rhs support confidence lift count  
## [1] {age=teens} => {income=(5.01e+03,2.44e+04]} 0.0617 1.00 2.1 37  
## [2] {income=(4.38e+04,6.31e+04]} => {save\_act=save\_act=YES} 0.1333 1.00 1.4 80  
## [3] {age=twenties} => {income=(5.01e+03,2.44e+04]} 0.1867 0.94 2.0 112  
## [4] {age=teens,   
## region=SUBURBAN} => {income=(5.01e+03,2.44e+04]} 0.0067 1.00 2.1 4  
## [5] {age=teens,   
## region=SUBURBAN} => {car=car=NO} 0.0067 1.00 2.0 4

## Generate rules and explore

As the test run reveals that the dataset is processed adequeatly, it can now be explored

rules <- apriori(bd, parameter = list(supp = 0.001, conf = 0.8))

## Apriori  
##   
## Parameter specification:  
## confidence minval smax arem aval originalSupport maxtime support minlen  
## 0.8 0.1 1 none FALSE TRUE 5 0.001 1  
## maxlen target ext  
## 10 rules FALSE  
##   
## Algorithmic control:  
## filter tree heap memopt load sort verbose  
## 0.1 TRUE TRUE FALSE TRUE 2 TRUE  
##   
## Absolute minimum support count: 0   
##   
## set item appearances ...[0 item(s)] done [0.00s].  
## set transactions ...[31 item(s), 600 transaction(s)] done [0.00s].  
## sorting and recoding items ... [31 item(s)] done [0.00s].  
## creating transaction tree ... done [0.00s].  
## checking subsets of size 1 2 3 4 5 6 7 8 9 10

## Warning in apriori(bd, parameter = list(supp = 0.001, conf = 0.8)): Mining  
## stopped (maxlen reached). Only patterns up to a length of 10 returned!

## done [0.11s].  
## writing ... [945484 rule(s)] done [0.16s].  
## creating S4 object ... done [0.45s].

options(digits = 2)  
inspect(rules[1:5])

## lhs rhs support confidence lift count  
## [1] {age=teens} => {income=(5.01e+03,2.44e+04]} 0.062 1.00 2.1 37  
## [2] {age=teens} => {current\_act=current\_act=YES} 0.055 0.89 1.2 33  
## [3] {region=SUBURBAN} => {current\_act=current\_act=YES} 0.083 0.81 1.1 50  
## [4] {children=3} => {pep=pep=NO} 0.092 0.81 1.5 55  
## [5] {income=(4.38e+04,6.31e+04]} => {save\_act=save\_act=YES} 0.133 1.00 1.4 80

The first 5 outcome revelas several information under the conditions that Support is 0.001, Confidence above 0.8  
“age=teen” is associated with income=(5.01e+03,2.44e+04] \*\*lowest width, and current\_act=current\_act=YES. “region=SUBURBAN” is associated with current\_act=current\_act=YES  
This is interesting for a company to note that “children=3” is associated with pep=pep=NO, with support of 0.092, confidence 0.81, lift 1.5

## Rule with High confidence

rules <- sort(rules,by='confidence',decreasing=TRUE)  
inspect(rules[1:5])

## lhs rhs support confidence lift count  
## [1] {age=teens} => {income=(5.01e+03,2.44e+04]} 0.0617 1 2.1 37  
## [2] {income=(4.38e+04,6.31e+04]} => {save\_act=save\_act=YES} 0.1333 1 1.4 80  
## [3] {age=teens,   
## region=SUBURBAN} => {income=(5.01e+03,2.44e+04]} 0.0067 1 2.1 4  
## [4] {age=teens,   
## region=SUBURBAN} => {car=car=NO} 0.0067 1 2.0 4  
## [5] {age=teens,   
## region=SUBURBAN} => {current\_act=current\_act=YES} 0.0067 1 1.3 4

Above rule outputs top five assocation with the highest confidence, mostly about “age=teens,region=SUBURBAN” associated with income=(5.01e+03,2.44e+04], car=car=NO, current\_act=current\_act=YES This result reveals informaiton about “age=teens,region=SUBURBAN” that not suprisingly the income bracket is low, car=NO, and current\_act=current\_act=YES

## Rule lhs pep=pep=YES

rules <- apriori(data=bd, parameter=list(supp=0.001, conf=0.08), appearance=list(default='rhs',lhs='pep=pep=YES'),control=list(verbose=F))  
inspect(rules[1:5])

## lhs rhs support confidence lift count  
## [1] {} => {region=SUBURBAN} 0.10 0.10 1 62   
## [2] {} => {children=3} 0.11 0.11 1 68   
## [3] {} => {income=(4.38e+04,6.31e+04]} 0.13 0.13 1 80   
## [4] {} => {age=old} 0.15 0.15 1 90   
## [5] {} => {region=RURAL} 0.16 0.16 1 96

The result reveals lift 1 for the top five outputs. Lift 1 implay that the probability of occurance of the antecedent and that of the consequent are independent of each other. pep=YES and the variables region=SUBURBAN, children=3, income=(4.38e+04,6.31e+04], age=old, region=RURAL are independent.

rules <- apriori(dat=bd, parameter=list(supp=0.001,conf=0.08, minlen=2),appearance=list(default='rhs',lhs='pep=pep=YES'), control=list(verbose=F))  
inspect(rules[1:5])

## lhs rhs support confidence lift  
## [1] {pep=pep=YES} => {region=SUBURBAN} 0.057 0.12 1.2   
## [2] {pep=pep=YES} => {income=(4.38e+04,6.31e+04]} 0.090 0.20 1.5   
## [3] {pep=pep=YES} => {age=old} 0.090 0.20 1.3   
## [4] {pep=pep=YES} => {region=RURAL} 0.077 0.17 1.0   
## [5] {pep=pep=YES} => {age=fifties} 0.090 0.20 1.2   
## count  
## [1] 34   
## [2] 54   
## [3] 54   
## [4] 46   
## [5] 54

Conclusion

The result shows that there is a high lift where variables on each side are dependent, however showing low support and low confidence. Interpreation is that customers who bought the product, PEP has a customer profile as suburban region, income level 3, old age bracket, also rural region, and age bracket fitities.