Doina Covaliu. Collaborative filltering Recommender Systems for Santander Bank

output: html_document: default word_document: default pdf_document: default —

#install the necesarry packages

```
#install.packages("dplyr", repos = "http://cran.us.r-project.org")
library("dplyr")
```

```
##
## Attaching package: 'dplyr'
```

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

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

```
#install.packages("ggplot2", repos = "http://cran.us.r-project.org")
library("ggplot2")
#install.packages("gplots",repos = "http://cran.us.r-project.org")
library("gplots")
```

```
##
## Attaching package: 'gplots'
```

```
## The following object is masked from 'package:stats':
##
##     lowess
```

```
#install.packages("ggcorrplot",repos = "http://cran.us.r-project.org")
library("ggcorrplot")
#install.packages("simputation", repos = "http://cran.us.r-project.org")
library("simputation")
#install.packages("wesanderson", repos = "http://cran.us.r-project.org")
#library("wesanderson")
#install.packages("recommenderlab", repos = "http://cran.us.r-project.org")
library("recommenderlab")
```

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

```
## Loading required package: arules
```

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

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

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

```
## Loading required package: proxy
```

```
##
## Attaching package: 'proxy'
```

```
## The following object is masked from 'package:Matrix':
##
##     as.matrix
```

```
## The following objects are masked from 'package:stats':
##
##     as.dist, dist
```

```
## The following object is masked from 'package:base':
##
##     as.matrix
```

```
## Loading required package: registry
```

```
#install.packages("arules", repos = "http://cran.us.r-project.org")
library("arules")
#install.packages("Matrix", repos = "http://cran.us.r-project.org")
library("Matrix")
#install.packages("reshape2", repos = "http://cran.us.r-project.org")
library("reshape2")
#install.packages("forecast", repos = "http://cran.us.r-project.org")
#library("forecast")
library("data.table")
```

```
##
## Attaching package: 'data.table'
```

```
## The following objects are masked from 'package:reshape2':
##
##     dcast, melt
```

```
## The following objects are masked from 'package:dplyr':
##
##     between, first, last
```

#read the csv file into a dataframe

```
data <- read.csv("C:/Users/Doina/Desktop/santander/train_santander.csv", stringsAsFactors = FALS
E)
```

Get the names of the colomn which are in spanish and replace them with the english equivalent

```
colnames(data)=c("fecha_dato"="Date","ncodpers"="Id","ind_empleado"="Emp_status","pais_residenci
a"="Country","sexo"="Sex","age"="Age", "fecha_alta"= "Date2", "ind_nuevo"="New_Customer", "antig
uedad"="Seniority","indrel"="Primary_customer", "ult_fec_cli_1t"="Pr_customer_ld", "indrel_1mes"
="Customer_type", "tiprel_1mes"="Customer_st_end_m", "indresi"="Residency", "indext"="Foreigner"
, "conyuemp"="emp_spouse", "canal_entrada"="Entry_chanal", "indfall"="Deceased", "tipodom"="Addr
ess_type", "cod_prov"="Province_code",   "nomprov"="province_name", "ind_actividad_cliente"="Act
ivity_st", "renta"="Household", "segmento"="Segment", "ind_ahor_fin_ult1"="Savings", "ind_aval_f
in_ult1"="Guarantees", "ind_cco_fin_ult1"="Current_Acc", "ind_cder_fin_ult1"="Derivada", "ind_cn
o_fin_ult1"="Payroll_Acc", "ind_ctju_fin_ult1"="Junior_Acc", "ind_ctma_fin_ult1"="M_particular_A
cc ",   "ind_ctop_fin_ult1"="Particular_Acc", "ind_ctpp_fin_ult1"="Particular_Plus_Acc", "ind_de
co_fin_ult1"="Short_term_dep", "ind_deme_fin_ult1"="Medium_term_dep",  "ind_dela_fin_ult1"="Long
_term_dep", "ind_ecue_fin_ult1"="e-account", "ind_fond_fin_ult1"="Funds", "ind_hip_fin_ult1" =
"Mortgage", "ind_plan_fin_ult1"="Pensions_Acc", "ind_pres_fin_ult1"= "Loans", "ind_reca_fin_ult
1"="Taxes", "ind_tjcr_fin_ult1"="Credit_Card", "ind_valo_fin_ult1"="Securities", "ind_viv_fin_ul
t1"="Home_Acc", "ind_nomina_ult1"="Payroll", "ind_nom_pens_ult1"="Pensions", "ind_recibo_ult1"=
"Direct_Debit")
```

As the database is very large we will look at 3 monthw of data from October 2015 untill December 2015 to predict what products should be recommended to active customers

```
train.data<-subset(data, data$Date=="2015-10-28"|data$Date=="2015-11-28"|data$Date=="2015-12-28"
)
```

Explore the structure of the data frame to understand the attributes, the class of the atributes

```
head(train.data)
```

```
##               Date      Id Emp_status Country Sex Age      Date2
## 6314953 2015-10-28 1217174          N      ES   V  22 2013-11-08
## 6314954 2015-10-28 1217176          N      ES   V  32 2013-11-08
## 6314955 2015-10-28 1217173          N      ES   H  23 2013-11-08
## 6314956 2015-10-28 1217172          N      ES   H  32 2013-11-08
## 6314957 2015-10-28 1217171          N      ES   V  25 2013-11-08
## 6314958 2015-10-28 1217170          N      ES   V  22 2013-11-08
##         New_Customer Seniority Primary_customer Pr_customer_ld
## 6314953            0        23                1               
## 6314954            0        23                1               
## 6314955            0        23                1               
## 6314956            0        23                1               
## 6314957            0        23                1               
## 6314958            0        23                1               
##         Customer_type Customer_st_end_m Residency Foreigner emp_spouse
## 6314953           1.0                 A         S         N           
## 6314954           1.0                 I         S         N           
## 6314955           1.0                 I         S         N           
## 6314956           1.0                 A         S         N           
## 6314957           1.0                 I         S         N           
## 6314958           1.0                 I         S         N           
##         Entry_chanal Deceased Address_type Province_code province_name
## 6314953          KHE        N            1            36    PONTEVEDRA
## 6314954          KHE        N            1            36    PONTEVEDRA
## 6314955          KHE        N            1            28        MADRID
## 6314956          KHE        N            1            28        MADRID
## 6314957          KHE        N            1            41       SEVILLA
## 6314958          KHE        N            1            15    CORUÃ‘A, A
##         Activity_st Household           Segment Savings Guarantees
## 6314953           1 222431.64 03 - UNIVERSITARIO       0          0
## 6314954           1 111080.34 03 - UNIVERSITARIO       0          0
## 6314955           0  45486.66 03 - UNIVERSITARIO       0          0
## 6314956           1  47477.85 03 - UNIVERSITARIO       0          0
## 6314957           0  51985.38 03 - UNIVERSITARIO       0          0
## 6314958           0  89040.69 03 - UNIVERSITARIO       0          0
##         Current_Acc Derivada Payroll_Acc Junior_Acc M_particular_Acc
## 6314953           1        0           0          0                0
## 6314954           1        0           0          0                0
## 6314955           0        0           0          0                0
## 6314956           0        0           1          0                0
## 6314957           1        0           0          0                0
## 6314958           1        0           0          0                0
##         Particular_Acc Particular_Plus_Acc Short_term_dep Medium_term_dep
## 6314953              0                   0              0               0
## 6314954              0                   0              0               0
## 6314955              0                   0              0               0
## 6314956              0                   0              0               0
## 6314957              0                   0              0               0
## 6314958              0                   0              0               0
##         Long_term_dep e-account Funds Mortgage Pensions_Acc Loans Taxes
## 6314953             0         0     0        0            0     0     0
## 6314954             0         0     0        0            0     0     0
## 6314955             0         0     0        0            0     0     0
```

```
## 6314956            0         0     0        0           0     0     1
## 6314957            0         0     0        0           0     0     0
## 6314958            0         0     0        0           0     0     0
##         Credit_Card Securities Home_Acc Payroll Pensions Direct_Debit
## 6314953           0          0        0       0        0            0
## 6314954           0          0        0       1        1            0
## 6314955           0          0        0       0        0            0
## 6314956           0          0        0       1        1            1
## 6314957           0          0        0       0        0            0
## 6314958           0          0        0       0        0            0
```

```
str(train.data)
```

```
## 'data.frame':    2710381 obs. of  48 variables:
##  $ Date              : chr  "2015-10-28" "2015-10-28" "2015-10-28" "2015-10-28" ...
##  $ Id                : int  1217174 1217176 1217173 1217172 1217171 1217170 1217175 1217169 1217168 1217205 ...
##  $ Emp_status        : chr  "N" "N" "N" "N" ...
##  $ Country           : chr  "ES" "ES" "ES" "ES" ...
##  $ Sex               : chr  "V" "V" "H" "H" ...
##  $ Age               : chr  " 22" " 32" " 23" " 32" ...
##  $ Date2             : chr  "2013-11-08" "2013-11-08" "2013-11-08" "2013-11-08" ...
##  $ New_Customer      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Seniority         : chr  "     23" "     23" "     23" "     23" ...
##  $ Primary_customer  : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ Pr_customer_ld    : chr  "" "" "" "" ...
##  $ Customer_type     : chr  "1.0" "1.0" "1.0" "1.0" ...
##  $ Customer_st_end_m : chr  "A" "I" "I" "A" ...
##  $ Residency         : chr  "S" "S" "S" "S" ...
##  $ Foreigner         : chr  "N" "N" "N" "N" ...
##  $ emp_spouse        : chr  "" "" "" "" ...
##  $ Entry_chanal      : chr  "KHE" "KHE" "KHE" "KHE" ...
##  $ Deceased          : chr  "N" "N" "N" "N" ...
##  $ Address_type      : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ Province_code     : int  36 36 28 28 41 15 28 28 36 28 ...
##  $ province_name     : chr  "PONTEVEDRA" "PONTEVEDRA" "MADRID" "MADRID" ...
##  $ Activity_st       : int  1 1 0 1 0 0 0 0 1 0 ...
##  $ Household         : num  222432 111080 45487 47478 51985 ...
##  $ Segment           : chr  "03 - UNIVERSITARIO" "03 - UNIVERSITARIO" "03 - UNIVERSITARIO" "03 - UNIVERSITARIO" ...
##  $ Savings           : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Guarantees        : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Current_Acc       : int  1 1 0 0 1 1 1 0 1 1 ...
##  $ Derivada          : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Payroll_Acc       : int  0 0 0 1 0 0 0 0 0 0 ...
##  $ Junior_Acc        : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ M_particular_Acc  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Particular_Acc    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Particular_Plus_Acc: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Short_term_dep    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Medium_term_dep   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Long_term_dep     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ e-account         : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Funds             : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Mortgage          : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Pensions_Acc      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Loans             : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Taxes             : int  0 0 0 1 0 0 0 0 0 0 ...
##  $ Credit_Card       : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Securities        : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Home_Acc          : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Payroll           : int  0 1 0 1 0 0 0 0 0 0 ...
##  $ Pensions          : int  0 1 0 1 0 0 0 0 0 0 ...
##  $ Direct_Debit      : int  0 0 0 1 0 0 0 0 1 0 ...
```

Check how many unique cutomers are there in the database. the second column hold the identification numbers for the customers

```
length(unique(train.data[,2]))
```

```
## [1] 915898
```

Check HOW many missing values are in the dataframe for each attribute

```
sapply(train.data, function(x) sum(is.na(x)))
```

```
##               Date                 Id         Emp_status
##                  0                  0                  0
##            Country                Sex                Age
##                  0                  0                  0
##              Date2       New_Customer          Seniority
##                  0                  0                  0
##   Primary_customer     Pr_customer_ld      Customer_type
##                  0                  0                  0
##   Customer_st_end_m          Residency          Foreigner
##                  0                  0                  0
##         emp_spouse       Entry_chanal           Deceased
##                  0                  0                  0
##       Address_type      Province_code      province_name
##                  1              11932                  0
##        Activity_st          Household            Segment
##                  0             586641                  0
##            Savings         Guarantees        Current_Acc
##                  0                  0                  0
##           Derivada        Payroll_Acc         Junior_Acc
##                  0                  0                  0
##   M_particular_Acc      Particular_Acc Particular_Plus_Acc
##                  0                  0                  0
##     Short_term_dep    Medium_term_dep      Long_term_dep
##                  0                  0                  0
##          e-account              Funds           Mortgage
##                  0                  0                  0
##       Pensions_Acc              Loans              Taxes
##                  0                  0                  0
##        Credit_Card         Securities           Home_Acc
##                  0                  0                  0
##            Payroll           Pensions       Direct_Debit
##                  0                  0                  0
```

Replace missing value of Household income with the mean household income of the customer’s province .

```
clean_data<-impute_proxy(train.data, Household ~ mean(Household,na.rm=TRUE) | province_name)
```

Province_code has a 3992 missing values, but the column is not needed because the same information is provided by province_name, as a result the Province_code column will be removed

```
clean_data<-clean_data[-20]
```

All the missing value were replaced with the approprite values, but there are some empty spaces for different variables, which need to be replaced. Emp_spouse has the value of “S” if the customer is the spouse of an employee and “N” otherwise. We notice that out of 2.710.381 observations, we have only 3 entry for S and 341 for N. The rest are blank spaces. In conclusion, the variable emp_spouse doesn’t offer a lot of information and the entire column will be removed.

```
#emp_spouce with value S
length(clean_data$emp_spouse[clean_data$emp_spouse=="S"])
```

```
## [1] 3
```

```
#emp_spouce with value N"
length(clean_data$emp_spouse[clean_data$emp_spouse=="N"])
```

```
## [1] 341
```

```
#blank spaces in  emp_spouce
length(clean_data$emp_spouse[clean_data$emp_spouse==""])
```

```
## [1] 2710037
```

```
clean_data<-clean_data[-16]
```

Sex column has 15 empty spaces and they will be replaced with the most comun value

```
#Number of blank space in Sex Column
length(clean_data$Sex[clean_data$Sex==""])
```

```
## [1] 15
```

```
# creating a function to calculate the mode
calculate_mode<-function(x){
  uniq<-unique(na.omit(x))
  uniq[which.max(tabulate(match(x,uniq)))]
}
clean_data$Sex[clean_data$Sex==""]<-calculate_mode(clean_data$Sex)
```

Pr_customer_Id has 2703492 empty space, as a result the column will be removed(as most of the cells were empty)

```
length(clean_data$Pr_customer_Id[clean_data$Pr_customer_Id==""])
```

```
## [1] 2703492
```

```
#remove the Pr_customer_Ld column
clean_data<-clean_data[-11]
```

The Customer_type column should have the following values:1,2,3,4 and P. We will replace 1.0 with 1, 2.0 with 2, 3.0 with 3 and 4.0 with 4, P with a value of 5 and the empty spaces will be replaced with the most comun value.

```
length(clean_data$Customer_type[clean_data$Customer_type==""])
```

```
## [1] 47325
```

```
unique(clean_data$Customer_type)
```

```
##  [1] "1.0" "1"   "3.0" "P"   "3"   ""    "2.0" "2"   "4.0" "4"
```

```
clean_data$Customer_type[clean_data$Customer_type=="P"]<-5
clean_data$Customer_type[clean_data$Customer_type=="1.0"]<-1
clean_data$Customer_type[clean_data$Customer_type=="2.0"]<-2
clean_data$Customer_type[clean_data$Customer_type=="3.0"]<-3
clean_data$Customer_type[clean_data$Customer_type=="4.0"]<-4
clean_data$Customer_type[clean_data$Customer_type==""]<-calculate_mode(clean_data$Customer_type)
clean_data$Customer_type<-as.factor(clean_data$Customer_type)
```

The 47325 empty spaces in Customer_st_end_m will be replaced as well as the most comun value

```
#nr of blank spaces in Customer_st_end_m
length(clean_data$Customer_st_end_m[clean_data$Customer_st_end_m==""])
```

```
## [1] 47325
```

```
#impoute the most comun value
clean_data$Customer_st_end_m[clean_data$Customer_st_end_m==""]<-calculate_mode(clean_data$Custom
er_st_end_m)
```

The Entry_chanal variable has 58.026 blank spaces. They will be replaced with the most frequent value that occurs in the case of females and then we will do the same thing for males

```
length(clean_data$Entry_chanal[clean_data$Entry_chanal==""])
```

```
## [1] 58026
```

```
Entry_chanal_female=calculate_mode(clean_data$Entry_chanal[grepl("V",clean_data$Sex)])
clean_data$Entry_chanal[grepl("V",clean_data$Sex) & clean_data$Entry_chanal==""]=Entry_chanal_fe
male

Entry_chanal_male=calculate_mode(clean_data$Entry_chanal[grepl("H",clean_data$Sex)])
clean_data$Entry_chanal[grepl("H",clean_data$Sex) & clean_data$Entry_chanal==""]=Entry_chanal_ma
le

rm(Entry_chanal_female, Entry_chanal_male)
```

The blank spaces in segment variable will be considered as different segment that it will be named “Other”

```
length(clean_data$Segment[clean_data$Segment==""])
```

```
## [1] 58842
```

```
clean_data$Segment[clean_data$Segment==""]<-"Other"
```

The province_name variable has 11.932 blank spaces. After further investigation we notice that the customers for whom the province_name is blank, 19 of them are from Spain and the most comun value will be imputed and the rest come from other countries than Spain. We will impute the value “International” for the blank spaces in this case.

```
#Number of observation with blank space in the province_name column"
length(clean_data$province_name[clean_data$province_name==""])
```

```
## [1] 11932
```

```
#The country of the customers with blank space in the province_name column"
unique(clean_data$Country[clean_data$province_name==""])
```

```
##   [1] "GB" "MX" "BE" "US" "SE" "AR" "IE" "BR" "CH" "VE" "DE" "FR" "QA" "DO"
##  [15] "DJ" "IL" "JP" "CO" "RO" "PE" "PT" "IT" "EC" "RU" "PL" "GT" "GA" "MA"
##  [29] "NO" "SN" "MR" "CN" "NL" "UA" "IN" "BG" "CL" "HN" "PY" "FI" "CR" "NI"
##  [43] "TW" "AL" "MZ" "LT" "SV" "GR" "EE" "CZ" "AT" "CA" "JM" "HU" "ET" "SA"
##  [57] "ES" "CM" "LU" "CI" "NG" "CU" "SG" "SK" "KE" "TR" "AU" "BY" "UY" "TG"
##  [71] "MD" "AD" "BO" "TN" "PA" "HR" "ZA" "PR" "DK" "EG" "GQ" "GE" "BA" "HK"
##  [85] "MK" "LY" "KR" "PK" "DZ" "LB" "TH" "GH" "KH" "AE" "RS" "AO" "NZ" "MM"
##  [99] "PH" "KW" "VN" "GI" "OM" "CG" "LV" "ML" "GN" "GW" "ZW" "BZ" "KZ" "CF"
## [113] "IS" "CD" "SL" "GM" "BM"
```

```
#Customers with blank space in the province_name column from Spain"
length(clean_data$province_name[clean_data$province_name==""& clean_data$Country=="ES"])
```

```
## [1] 19
```

```
clean_data$province_name[clean_data$province_name==""& clean_data$Country=="ES"]<-calculate_mode
(clean_data$province_name)

clean_data$province_name[clean_data$province_name==""]<-"International"
```

Date2, the date at which the individual became a customer of the bank is not needed as the same information is reflected in the Seniority(months)= the difference between Date and Date 2

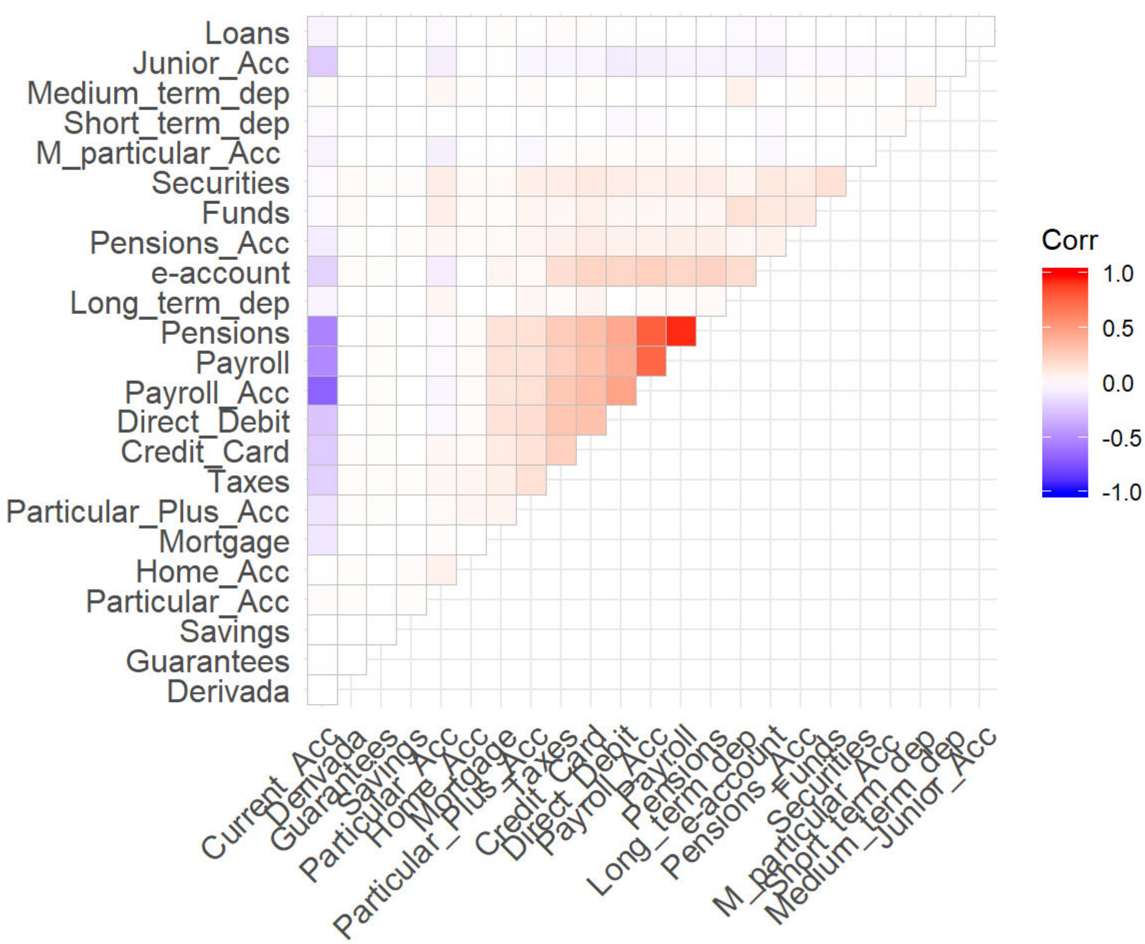
```
clean_data<-clean_data[-7]
```

The purpose of recommender system is to recommend new products to the active customers.As a result, inactive and deceased customers will be removed

```
#subsetting just the customers who are active
clean_data<-subset(clean_data, clean_data$Activity_st=="1")
#subsetting just the customers who are not deceased
clean_data<-subset(clean_data, clean_data$Deceased=="N")
```

Looking at the correlation between products, it is noticeable that there is a strong correlation between Pension and Payroll Acccount, between Pension and Direct Debit and Payroll_Acc and Direct Debit

```
correlation<-cor(clean_data[,21:44])
ggcorrplot(correlation, hc.order = TRUE, type = "upper")
```

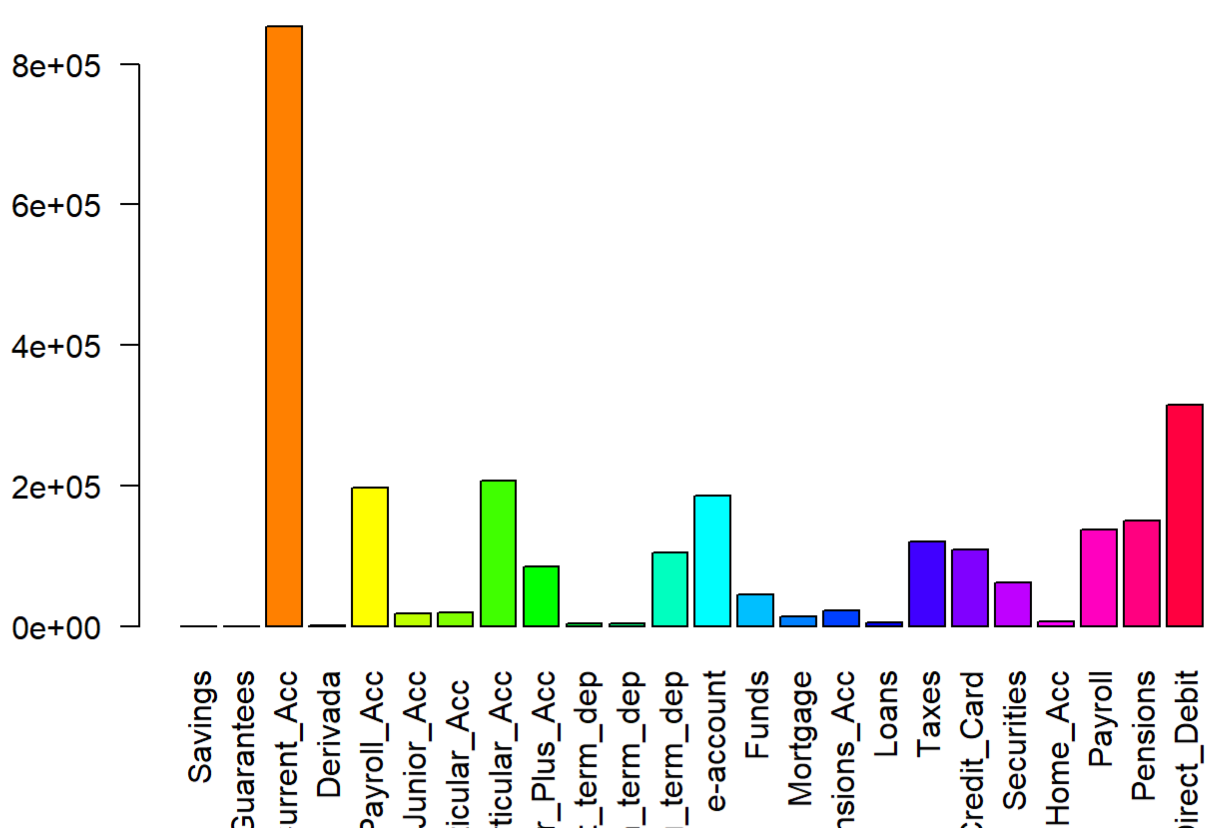


The most popular products accoring to the plot are: Curent Account, followed by Direct_Debit, Particular Account, e-account and Payroll Account

```
corr_data<-clean_data[,21:44]
x<-colSums(corr_data)
order((x),decreasing = TRUE)
```

```
##  [1]  3 24  8  5 13 23 22 18 19 12  9 20 14 16  7  6 15 21 17 10 11  4  1
## [24]  2
```

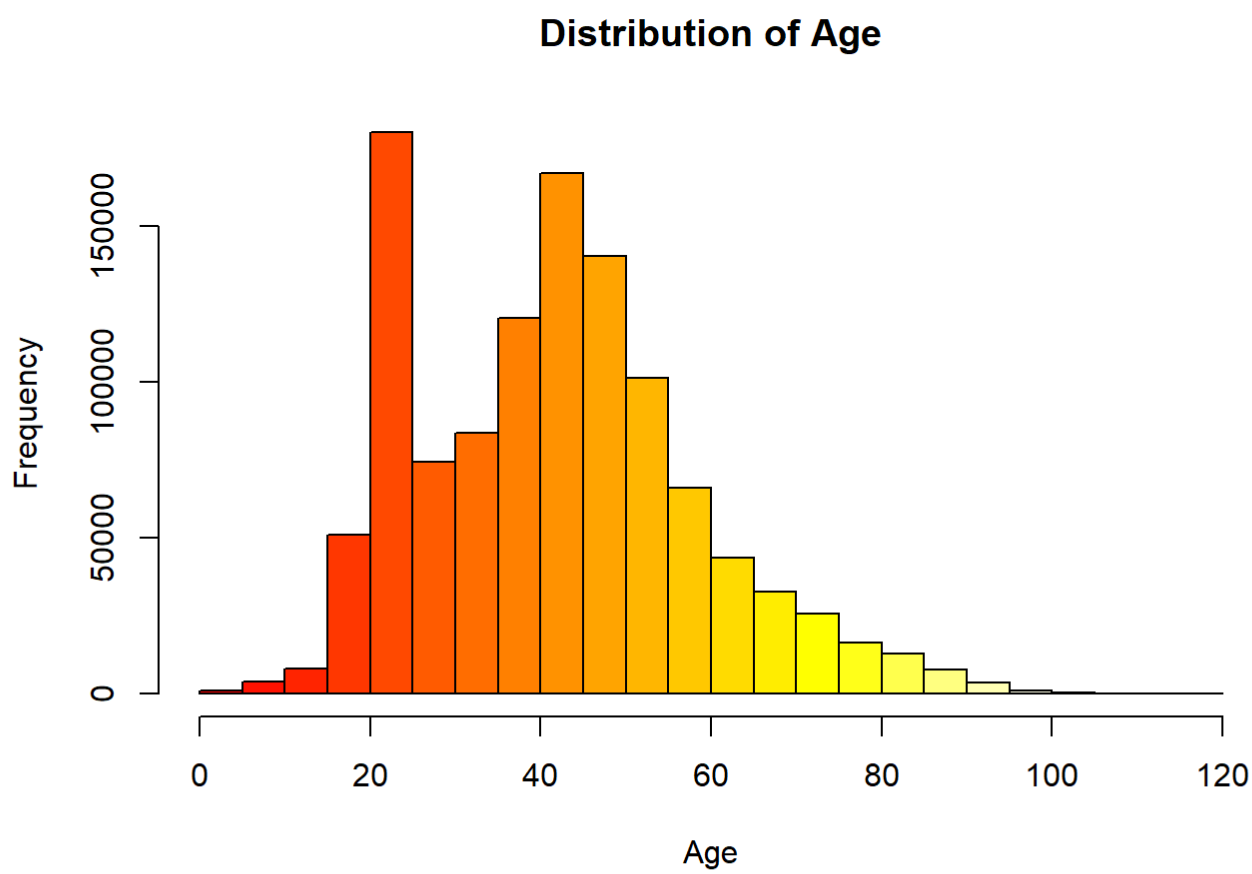
```
barplot(x, las=2,beside=TRue,col = rainbow(24))
```




```
rm(x, corr_data)
```

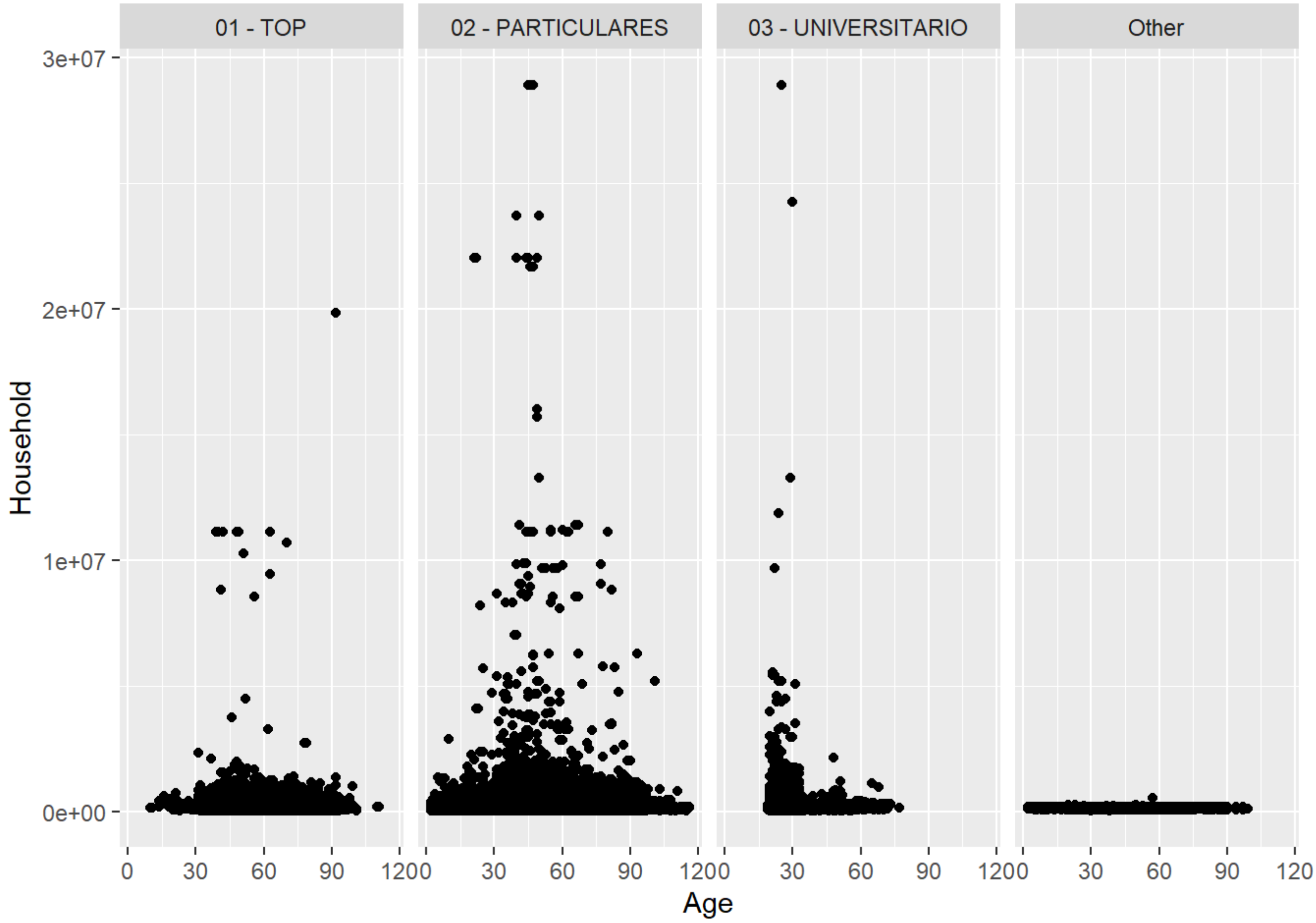
#Distribution of age of active customers

```
clean_data$Age<-as.numeric(as.character(clean_data$Age))
hist(clean_data$Age, col=heat.colors(20), main="Distribution of Age", xlab="Age")
```



Distribution of Household income per age per segment. we observe that PARTICULARES is the segment that has the most of the customer with higher household income, followed by UNIVERSITARIO

```
qplot(Age, Household, data = clean_data, facets = . ~Segment, )
```

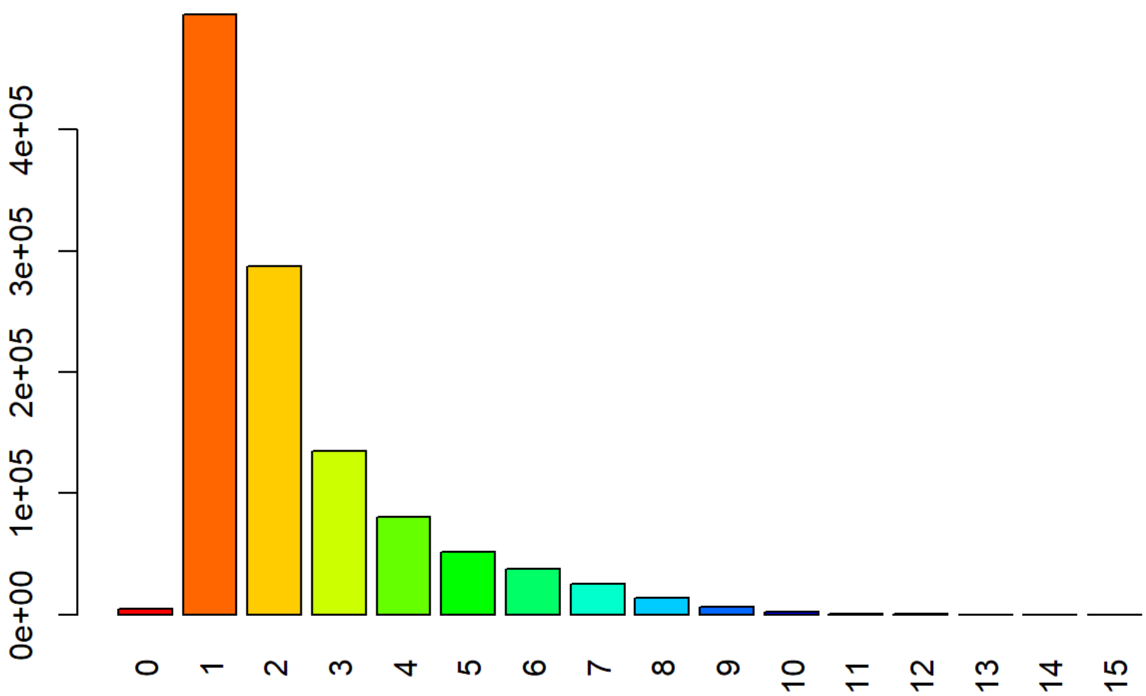


In order to build a Item-Based corellative filterying recommender system, a closer look at the products that customers have at the bank is needed. It appears that only one customer has 14 products at the same time and there are 7477 customers who don't have any products. This last category of customers don't have any product history and will not bring any information to the recommender system. Many of the customers (844443) have only one product. In the analysis only customers who have at least one product will be considered.

```
totalproducts<-rowSums(clean_data[,21:44], na.rm = TRUE)
table(totalproducts)
```

```
## totalproducts
##      0      1      2      3      4      5      6      7      8      9 
##   4511 495224 287032 134889  80852  51931  37850  25577  13856   6110 
##     10     11     12     13     14     15 
##   2355    822    202     50      3      1
```

```
barplot(table(totalproducts), las=3, col=rainbow(15))
```




```
rm(totalproducts)
```

Colaborative filtering Recommenders systems

Building the rating Matrix in the format accepted by recommenderlab package

```
# In order to use data.table package we transform the data frame into a data frame recognized by
data.table

S_dataset<-as.data.table(clean_data)

# extract labels for the products from the dataset
names_col = colnames(S_dataset[21:44])
names_products = names_col[21:44]


# we make sure to include in our model  just those customers who have at least one product at th
e bank
setkey(S_dataset, Id)
S_dataset = S_dataset[S_dataset[,rowSums(.SD, na.rm = TRUE), .SDcols=names_products]>0]

# The products are type integer and we will tranform it into type numeric
S_dataset= S_dataset[, .SD, .SDcols=c("Id", names_products)]
S_dataset= S_dataset[, (names_products):=lapply(.SD, as.numeric), .SDcols=names_products]



# create ratings matrix for the chosed Santander dataset
S_matrix = as.matrix(S_dataset[, .SD, .SDcols=names_products])
rownames(S_matrix) = S_dataset$Id
S_matrix = as(S_matrix, "binaryRatingMatrix")
S_matrix
```

```
## 1136754 x 24 rating matrix of class 'binaryRatingMatrix' with 2664718 ratings.
```

Create an evaluation scheme to evaluate the 3 models using “split” method , which separates the data into training set 70% and test set 30%.

```
eval = evaluationScheme(S_matrix, method="split", train=0.7, given=-1)
eval
```

```
## Evaluation scheme using all-but-1 items
## Method: 'split' with 1 run(s).
## Training set proportion: 0.700
## Good ratings: NA
## Data set: 1136754 x 24 rating matrix of class 'binaryRatingMatrix' with 2664718 ratings.
```

Builing the Item-Based Collaborative filtering recommender system and the predict function which will be used for evaluation

```
#create the recommender object for IBCF
rec <- Recommender(getData(eval, "train"), "IBCF", parameter = list(k=50))
rec
```

```
## Recommender of type 'IBCF' for 'binaryRatingMatrix'
## learned using 795727 users.
```

```
#use the predict function to obtain a list of recommendations for the train set which will be us
ed in the calcPredictionAccuracy to compare it with the list of recommendation for the test set
("unkown data")
pred <- predict(rec, getData(eval, "known"), type="topNList", n=5)
pred
```

```
## Recommendations as 'topNList' with n = 5 for 341027 users.
```

```
#evaluation of the IBCF recommender system on the test set(unknown observations) and obtaining t
he evaluation metrics
eval_IBCF<- calcPredictionAccuracy(pred, getData(eval, "unknown"), given = -1)
eval_IBCF<-as.data.frame(as.list(eval_IBCF))

#calculate the MAE and Accuracy
eval_IBCF$MAE = (eval_IBCF$FP+eval_IBCF$FN)/(eval_IBCF$TN+eval_IBCF$FN+eval_IBCF$FP+eval_IBCF$T
P)

eval_IBCF$Accuracy=(eval_IBCF$TP+eval_IBCF$TN)/(eval_IBCF$TN+eval_IBCF$FN+eval_IBCF$FP+eval_IBCF
$TP)
```

Building the Random Recommenders systems and obtaining the vealuation metrics

```
#create the recommender object
rec2<- Recommender(getData(eval, "train"), "RANDOM", parameter = NULL)

#use the predict function to obtain a list of recommendations for the train set which will be us
ed in the calcPredictionAccuracy to compare it with the list of recommendation for the test set
("unkown data")
rec2
```

```
## Recommender of type 'RANDOM' for 'binaryRatingMatrix'
## learned using 795727 users.
```

```
pred2 <- predict(rec2, getData(eval, "known"), type="topNList", n=5)


#determine the evaluation metrics of the recommender system

eval_Random<- calcPredictionAccuracy(pred2, getData(eval, "unknown"), given = -1)
eval_Random<-as.data.frame(as.list(eval_Random))

#calculate MAE and Accuracy
eval_Random$MAE = (eval_Random$FP+eval_Random$FN)/(eval_Random$TN+eval_Random$FN+eval_Random$FP+
eval_Random$TP)

eval_Random$Accuracy=(eval_Random$TP+eval_Random$TN)/(eval_Random$TN+eval_Random$FN+eval_Random
$FP+eval_Random$TP)
```

Building the Popular Recommender sytem and calculating the evaluation metrics

```
#create the recommender object
rec3<- Recommender(getData(eval, "train"), "POPULAR", parameter = NULL)
rec3
```

```
## Recommender of type 'POPULAR' for 'binaryRatingMatrix'
## learned using 795727 users.
```

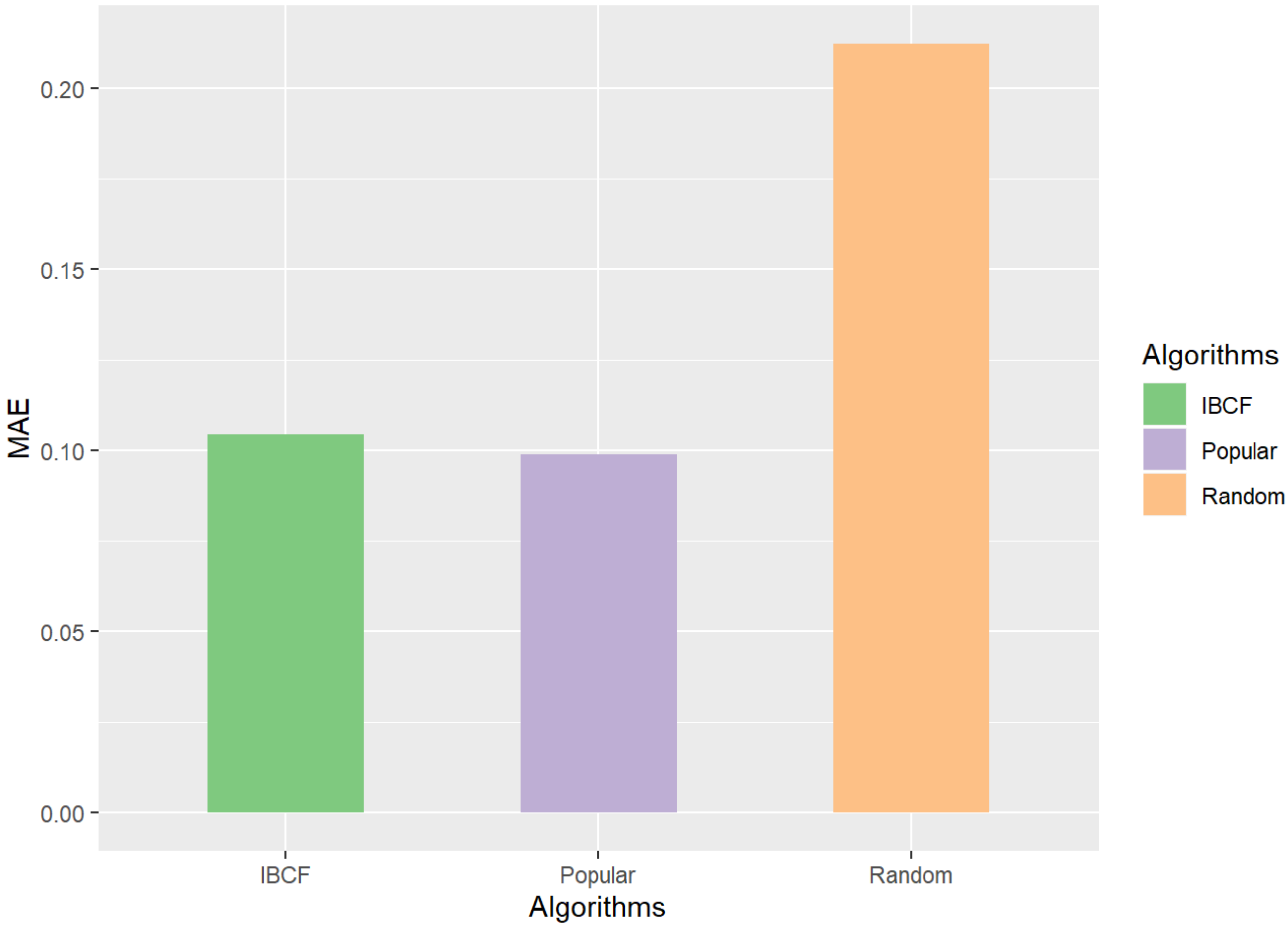
```
pred3 <- predict(rec3, getData(eval, "known"), type="topNList", n=5)

#determine the evaluation metrics
eval_Pop<- calcPredictionAccuracy(pred3, getData(eval, "unknown"), given = -1)
eval_Pop<-as.data.frame(as.list(eval_Pop))

#calculate mean absolute error
eval_Pop$MAE = (eval_Pop$FP+eval_Pop$FN)/(eval_Pop$TN+eval_Pop$FN+eval_Pop$FP+eval_Pop$TP)
#calculate accuracy
eval_Pop$Accuracy=(eval_Pop$TP+eval_Pop$TN)/(eval_Pop$TN+eval_Pop$FN+eval_Pop$FP+eval_Pop$TP)
```

#Comparing algorithms. it is clear that the Popular recommender system performs the best, followed by Item-based recommender system

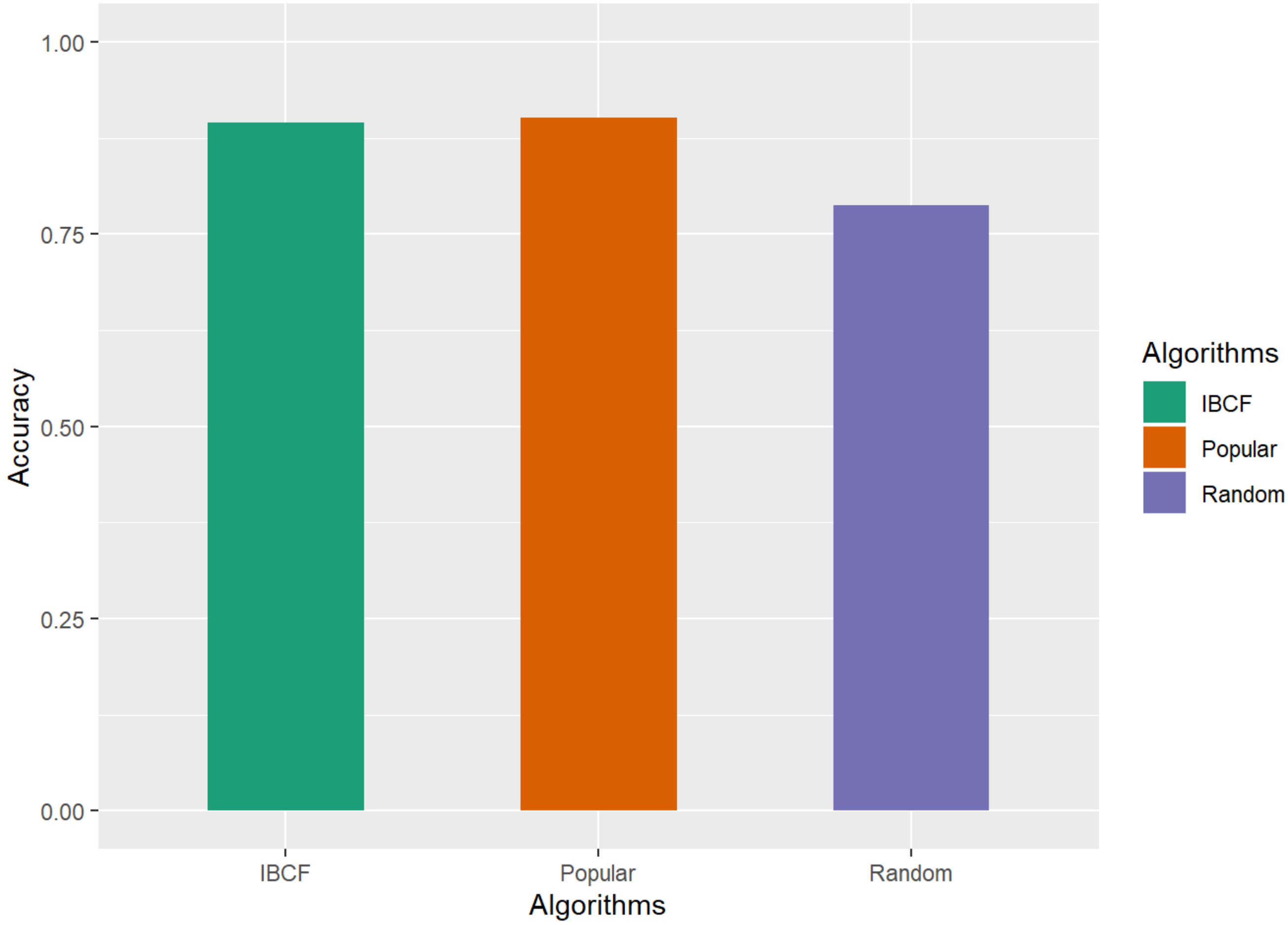
```
Results<-data.frame(Algorithms=c("IBCF", "Popular","Random"))
Results_Alg<-rbind(eval_IBCF,eval_Pop, eval_Random)
Results_Final<-cbind.data.frame(Results,Results_Alg)
Results_Final
```

```
##   Algorithms       TP       FP         FN       TN precision    recall
## 1       IBCF 0.386336 2.432253 0.1773818 22.00403 0.1370671 0.6853357
## 2    Popular 0.456046 2.362543 0.1076718 22.07374 0.1617994 0.8089969
## 3     Random 0.130626 4.869374 0.4330918 19.56691 0.0261252 0.2317224
##          TPR       FPR       MAE  Accuracy
## 1 0.6853357 0.1013439 0.1043854 0.8956146
## 2 0.8089969 0.0984393 0.0988086 0.9011914
## 3 0.2317224 0.1992549 0.2120986 0.7879014
```

```
ggplot(data=Results_Final, aes(x=Algorithms, y=MAE, fill=Algorithms)) + geom_bar(stat="identity"
, width = 0.5)+scale_fill_brewer(palette = "Accent")
```




```
ggplot(data=Results_Final, aes(x=Algorithms, y=Accuracy, fill=Algorithms)) + geom_bar(stat="iden
tity", width = 0.5)+ylim(0,1)+scale_fill_brewer(palette = "Dark2")
```



Ploting the ROC curve for all the algorithms, a easy way to undersand how the algorithms will perform for 1,5,10,15,20 or 24 item predicted.

```
#creating the evaluation scheme using cross validation method
scheme <- evaluationScheme(S_matrix, method="cross-validation", k=10, given=-1)
scheme
```

```
## Evaluation scheme using all-but-1 items
## Method: 'cross-validation' with 10 run(s).
## Good ratings: NA
## Data set: 1136754 x 24 rating matrix of class 'binaryRatingMatrix' with 2664718 ratings.
```

```
#create a list with all the algorithms
algorithms <- list("item-based CF" = list(name="IBCF", param=list(k=50)), "random items" = list
(name="RANDOM", param=NULL), "popular items" = list(name="POPULAR", param=NULL))
algorithms
```

```
## $`item-based CF`
## $`item-based CF`$name
## [1] "IBCF"
##
## $`item-based CF`$param
## $`item-based CF`$param$k
## [1] 50
##
##
##
## $`random items`
## $`random items`$name
## [1] "RANDOM"
##
## $`random items`$param
## NULL
##
##
## $`popular items`
## $`popular items`$name
## [1] "POPULAR"
##
## $`popular items`$param
## NULL
```

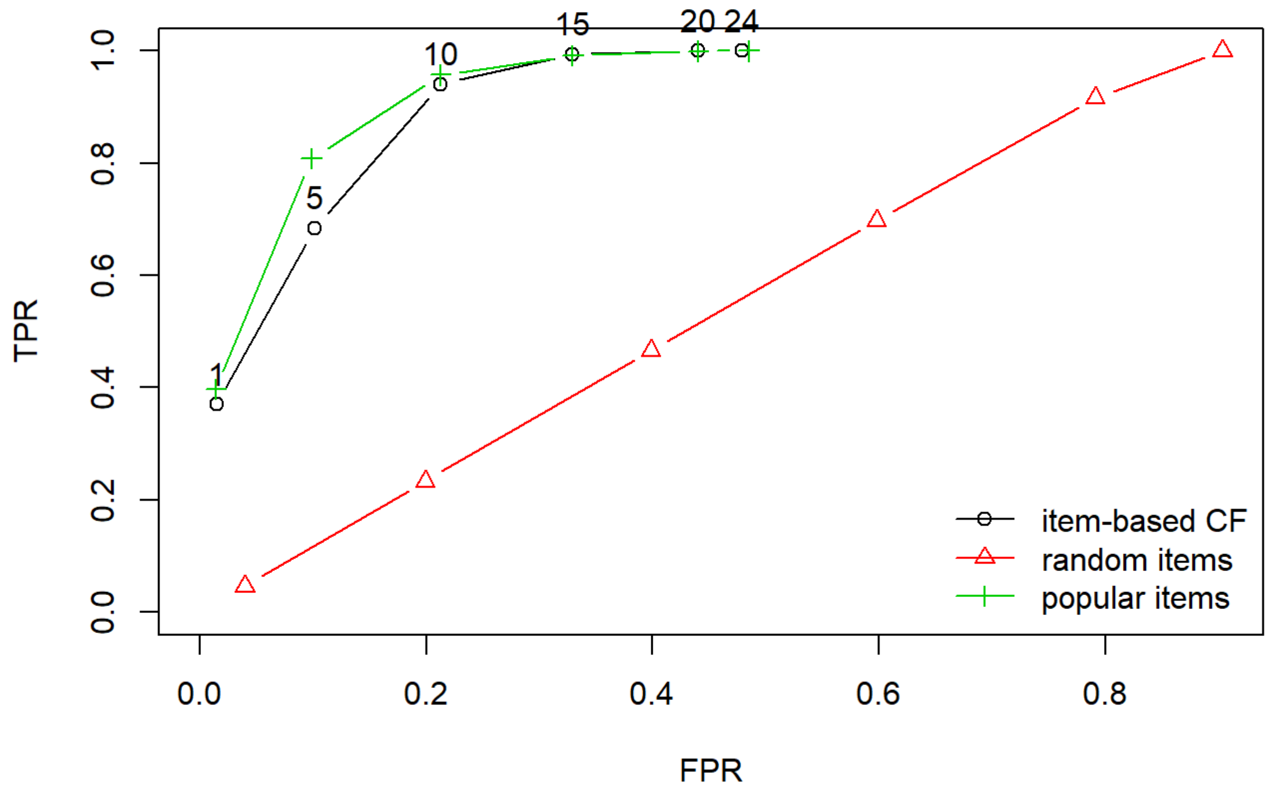
```
#obtain the evaluation metrics and plot them
results <- evaluate(scheme, algorithms, type = "topNList",n=c(1,5,10,15,20,24))
```

```
## IBCF run fold/sample [model time/prediction time]
##   1  [1.38sec/15.9sec]
##   2  [1.28sec/14.18sec]
##   3  [1.2sec/14.41sec]
##   4  [1.22sec/15.93sec]
##   5  [1.07sec/14.5sec]
##   6  [1.01sec/13.86sec]
##   7  [1.4sec/16.41sec]
##   8  [1.25sec/15.3sec]
##   9  [1.27sec/13.17sec]
##   10  [1.03sec/16.33sec]
## RANDOM run fold/sample [model time/prediction time]
##   1  [0sec/15.34sec]
##   2  [0sec/15.42sec]
##   3  [0sec/15.89sec]
##   4  [0sec/15.54sec]
##   5  [0sec/14.92sec]
##   6  [0sec/16.47sec]
##   7  [0sec/16.41sec]
##   8  [0sec/16.28sec]
##   9  [0sec/15.29sec]
##   10  [0sec/17.28sec]
## POPULAR run fold/sample [model time/prediction time]
##   1  [0.04sec/323.31sec]
##   2  [0.05sec/316.78sec]
##   3  [0.04sec/327.46sec]
##   4  [0.05sec/328.83sec]
##   5  [0.04sec/327.92sec]
##   6  [0.05sec/316.26sec]
##   7  [0.04sec/320.57sec]
##   8  [0.05sec/314.62sec]
##   9  [0.05sec/316.83sec]
##   10  [0.04sec/322.54sec]
```

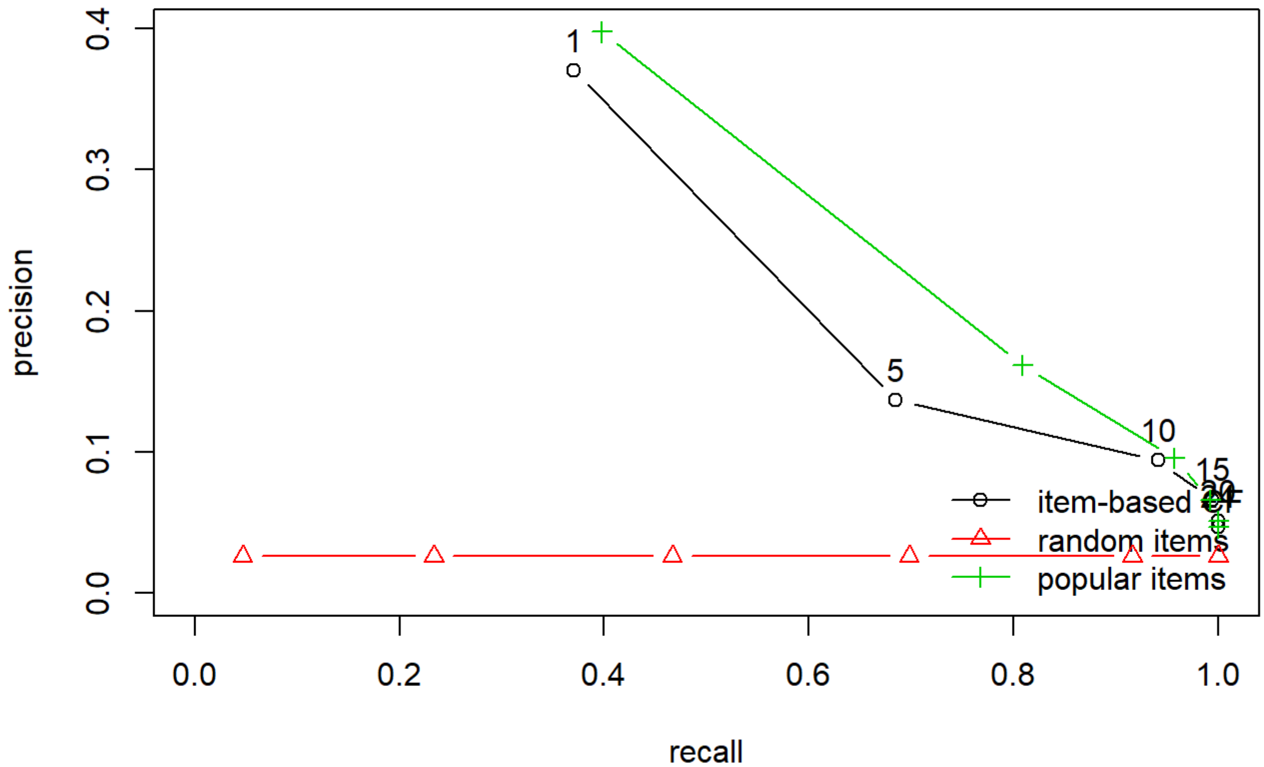
```
results
```

```
## List of evaluation results for 3 recommenders:
## Evaluation results for 10 folds/samples using method 'IBCF'.
## Evaluation results for 10 folds/samples using method 'RANDOM'.
## Evaluation results for 10 folds/samples using method 'POPULAR'.
```

```
plot(results, annotate=TRUE)
```




```
plot(results, "prec/rec", annotate=TRUE)
```



Display a list of recomendations for the first users

```
pred3_test<-predict(rec3, getData(eval, "unknown"), type="topNList", n=5)
predicted_items<-as(pred3_test,"list")
predicted_items[2:11]
```

```
## $`15892`
## [1] "Current_Acc"    "Direct_Debit"   "Particular_Acc" "Payroll_Acc"
## [5] "e-account"
##
## $`15892`
## [1] "Current_Acc"    "Particular_Acc" "Payroll_Acc"    "e-account"
## [5] "Pensions"
##
## $`15893`
## character(0)
##
## $`15895`
## [1] "Current_Acc"    "Direct_Debit"   "Particular_Acc" "Payroll_Acc"
## [5] "e-account"
##
## $`15895`
## [1] "Current_Acc"    "Direct_Debit"   "Particular_Acc" "Payroll_Acc"
## [5] "e-account"
##
## $`15896`
## character(0)
##
## $`15897`
## [1] "Current_Acc"    "Direct_Debit"   "Particular_Acc" "Payroll_Acc"
## [5] "e-account"
##
## $`15899`
## [1] "Current_Acc"  "Direct_Debit" "Payroll_Acc"  "e-account"
## [5] "Pensions"
##
## $`15900`
## [1] "Current_Acc"    "Particular_Acc" "Payroll_Acc"    "e-account"
## [5] "Pensions"
##
## $`15900`
## [1] "Current_Acc"    "Particular_Acc" "Payroll_Acc"    "e-account"
## [5] "Pensions"
```