Cafe Bazaar Report

Mohammad Mazraeh

May 14, 2016

Synposis

In this research we want to analyse user payment data and cluster users in order to plan for promotions or solutions to make more money (Increase users long term value)! User Segmentation Analysis and recommendations are usually done in two ways:

Find Hot Products in a group and recommend it to users

Recommend most probable interesting products to each user

In this research we will build both of recommendation system models. First method is done by finding top products in a particular group and recommend it to other groups. Second method is done using User Based Collaborative Filtering (UBCF) technique.

Data Processing

In this part first we generate some raw data and then in cleaning phase convert it to useful features to do our analysis.

Getting Data

In this part we would simulate some data as input for our analysis. We use some parameters to generate data. The parameters are:

n is number of data to generate.

numUsers is the number of users.

typeFactor is a factor including software types. in this analysis it's App or Game.

appCount: Number of apps in cafe bazaar.

```r
n <- 1000000
numUsers <- 10000
appCount <- 100
typeFactor <- as.factor(1:10)
# Set seed for reproducability
set.seed(789)
```

We assume the raw data table would have one ID and 4 features: ***userID***: unique user ID. ***cat***: the payment was for an Application or a Game ***appID***: A unique ID for each app ***amount***: the amount of payment ***payDate***: payment date

It is assumed that installations with no payment are recorderd as $amount = 0$. Also any payment (including in-app purchases) is recorderd in the table.

To generate dataset we followed these steps: For each record *userID* is sampled from user IDs. *cat* is app category.*appID* is a unique ID for each app which have been sampled randomly. for *amount* variable 0.9 of records will have zero value (Installations with no payment) and others are assumed to be from a normal distribution with arbitary mean and std. at last *payDate* is sampled randomly between "2013/01/01" and "2016/01/01".

```
payments <-
      data.table(userID = sample(1:numUsers,size = n,replace = TRUE),
                 cat = sample(typeFactor,
                              replace = TRUE,
                              prob = c(0.2,0.15,0.1,0.05,0.25,0.1,0.05,0.02,0.05,0.03)),
                 appID = sample(1:appCount,replace = TRUE),
                 amount = round(sample(c(abs(rnorm(n = 0.1*n,
                                                   mean = 3000,sd = 2000)),
                                         rep(0,0.9*n))),-2),
                 payDate = sample(seq(
                       as.Date('2013/01/01'),
                       as.Date('2016/01/01'),
                       by="day"), n,replace = TRUE))
```

Next we create a rating matrix. in *ratings* matrix each row corresponds to a user and each column corresponds to an app. Thus *ratings* matrix is an 10^{4} * 100 matrix. 0.9 of ratings are set to zero (and then replaced by NaN) which means user have not rated the apps.

```
# Ratings Matrix is Filled Randomly.
ratings = matrix(sample(0:5,numUsers*appCount,
                        replace = TRUE,
                        prob = c(0.9,0.02,0.02,0.02,0.02,0.02)),
                 nrow = numUsers)
ratings[ratings==0] <- NaN
```

Preprocess Data

One of the most useful techniques to analyse user purchase history is to use RFM Models. R,F and M stand for Recency,Frequency and Monetary respectively. In this part we want to convert our input data into RFM Features:

FirstPurchaseDate: Roughly shows how long the user is with us.

LastPurchaseDate: basis of Recency.

Numberofpayments: basis of Frequency.

CatCount: basis of breadth (Number of unique categories which user is involved)

TotalAmount: basis of Monetary.

```
# Building RFM Features
RFM <- payments  %>%
      group_by(userID)  %>%
      summarise(FirstPurchaseDate = min(payDate),
                LastPurchaseDate = max(payDate),
                NumberPayments = sum(amount > 0),
                NumberofApps = length(unique(appID)),
                Breadth = length(unique(cat)),
                TotalAmount = sum(amount))
```

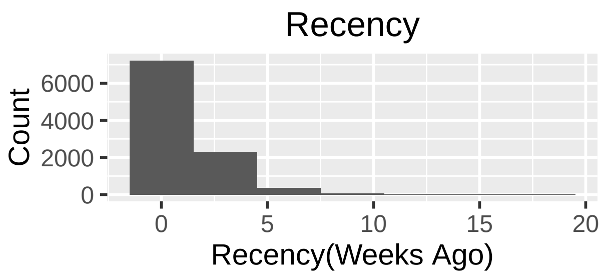
```r
LastDate <- max(RFM$LastPurchaseDate)
RFM <- RFM %>% mutate(R = as.numeric(LastDate-LastPurchaseDate),
                      PayRatio = NumberPayments / NumberofApps)
```

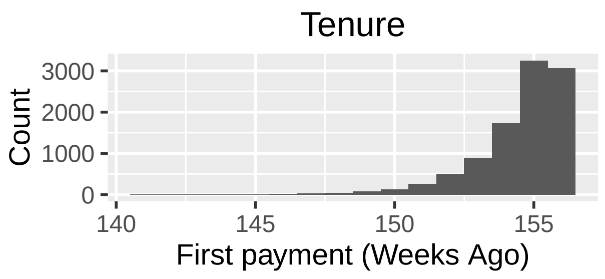
Analysis

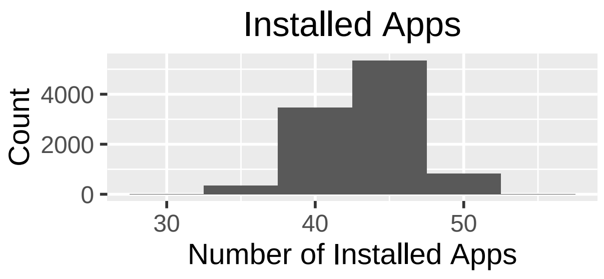
In this part we do the main part of report. First we look at data to get familliar with the fields and distributions. After that we do a simple user segmentation according to conventional RFM features and develop a simple topN recommender system to recommend 10 apps for some clusters. Finally we develope a per-user recommendation system using collaborative filtering.

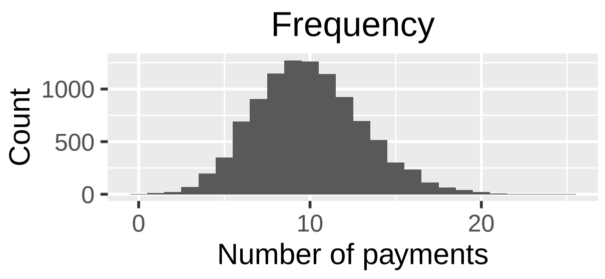
Exploratory Data Analysis

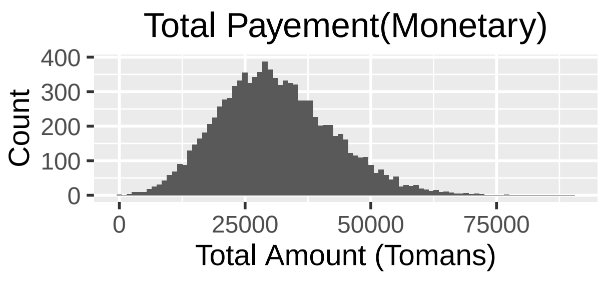
Like any other data analysis after getting and cleaning data we should do some exploratory data analysis to get familiar with data and hopefully it gives good information and clues to continue the research. Let's have each features histogram and discuss it:





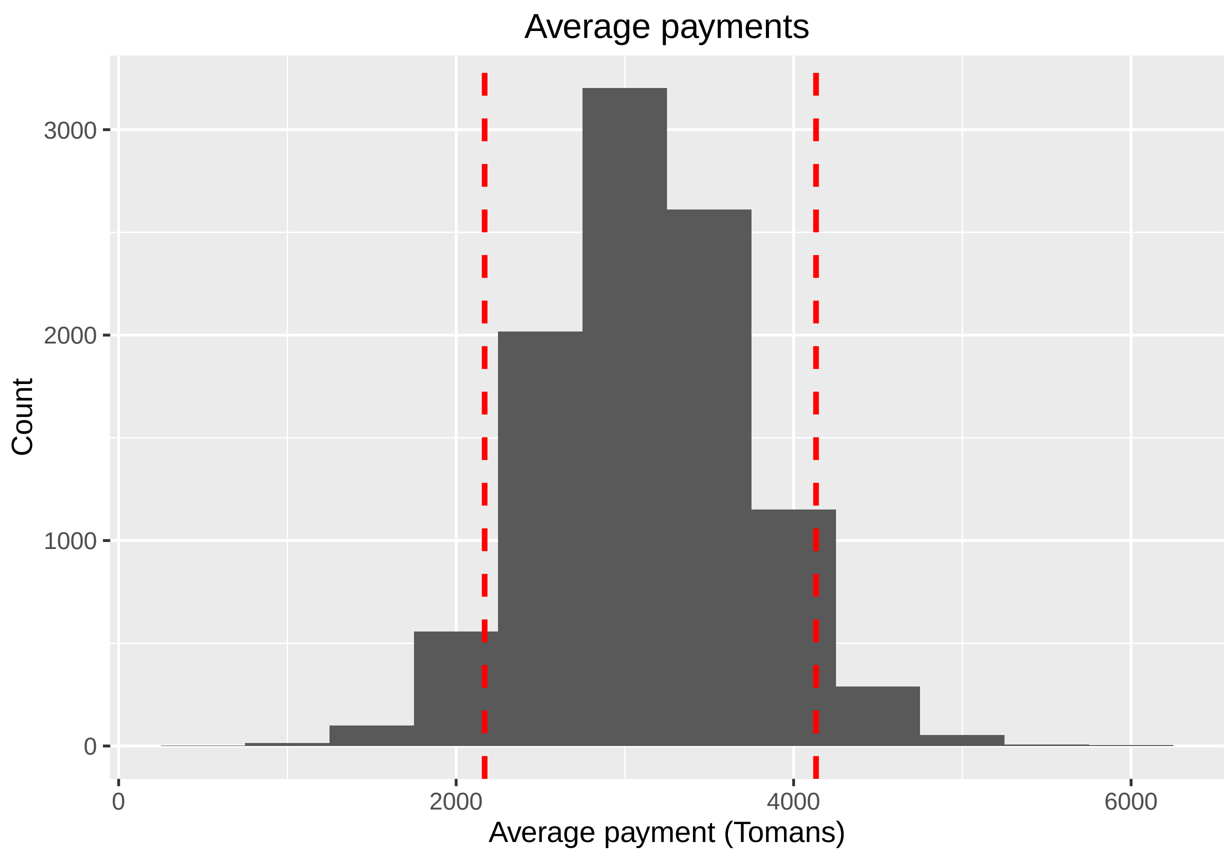






By looking at recency plot we see that last payments in most recent weeks are more than some weeks ago, which seems to be normal. In Tenure plot we see that there is no recent first purchase in our users which could have two meanings: 1- We have covered all possible users and hence we have not new user (Often wrong!) 2- We have a problem in our system to introduce ourself to new users!

Frequency plot gives us a clue that most users install between 35-50 apps from cafe bazaar and paied for 5-15 of them.In Monetary plot we see histogram for users total payments.This plot shows that most users spend an a total amount between 10000 and 50000 Tomans on different apps during their membership. It's good to see how users pay for apps in average.



This shows that 90% of users would pay an amount between 2169 and 4133 Tomans.

PayRatio feature indicates how often users pay in apps.

```
summary(RFM$PayRatio)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.1795  0.2250  0.2297  0.2750  0.5897
```

By looking to *PayRatio* summary it tells that mostly 0.2 of apps are get paied.

User Segmentation

In this part we want to segment users using RFM features. By looking at EDA histograms we can categorize each feature into some intervals.

By doing so we can have an overview of users RFM features

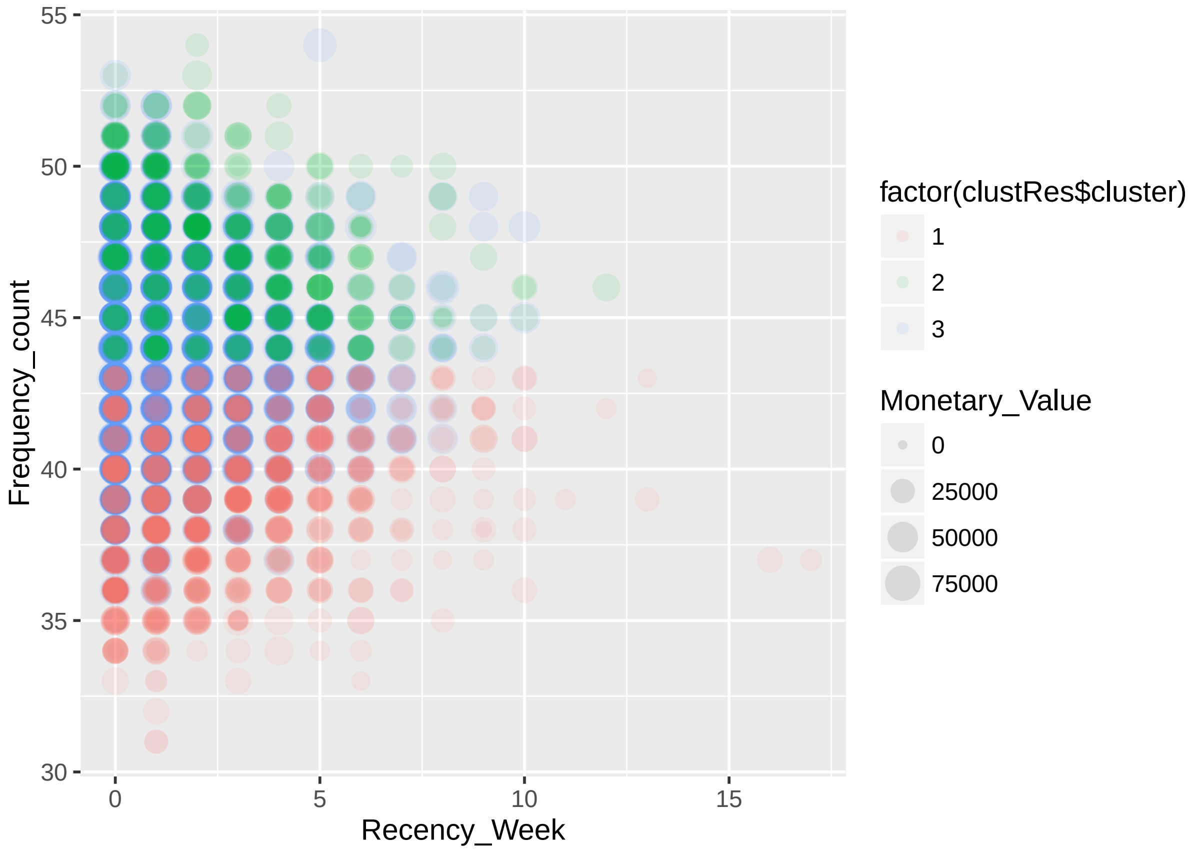
```
summary(select(RFM_Segs,Recency,Frequency,Monetary,Tenure))
```

```
##  Recency       Frequency       Monetary       Tenure
##  0-1 :7232   0-10   :    0   0-1K    :   2   0-140  :   2
##  1-4 :2317   10-50  :    0   1K-10K  : 180   140-150:   0
##  4-8 : 411   50-80  :10000   10K-50K:    0   150-155:   0
##  8-10:  33   80-100 :    0   10K-30K:4739   155+   :9998
##  10+ :   7   100-110:    0   30K-50K:4494
##              110+   :    0   50K+    : 585
```

In the above table we see that most users have a payement or installation at least in last 4 weeks. This means a little chance to have lost users! The Frequency confirms that users install 50-80 apps in their membership. The Monetary shows users total spend statistics (most likely between 10K and 50K Tomans). and the Tenure says we have no new user!

Group Recommendations

In this part we want to cluster users using RFM features and create some plans for customers which can be more involved and more benefitial in our syste, After cleaning data and get RFM features we use KMeans algorithm to cluster users. As mentioned before some number of clusters have been tested and K=3 selected as most interpretable cluster count. It's worth mentioning that each of RFM features have been scaled between 0 and 1 to prevent feature range effects.



In the above plot color shows user cluster, horizontal axes is Recency, vertical axes is Frequncy and the size of points shows users Monetary. Let's call users which have high frequency and most recent activities **good users**.

```
goodUsers <- RFM$userID[which(clustRes$cluster == 2)]
```

In the plot good users are in green area. TO suggest some apps to non-good user hoping they install the apps and pay for them we can identify hot apps in green area and suggest them to other users (or any other kind of plans for those apps). This is done by calculating app-rate average for good users and select top 10 ratings. After that we have 10 app IDs which can be used for any king of recommendation or promotion planning for non-good users.

```
appRates <- colMeans(ratings[goodUsers,],na.rm = TRUE)
top10 <- sort(appRates,decreasing = TRUE, index.return=TRUE)$ix[1:10]
print(top10)
```

```
##  [1] 58 48 30 16 93 68 44 80 17 36
```

This 10 app IDs are IDs for hot apps in good users cluster. we can recommend them to other users to get more installation and hopefully more payements.

In the next part we are going to build a per-user recommendation model.

Per User Recommendations

If we have possibility to recommend apps per user, we can use users rating analysis to recommend most probable apps to the users. One of the most conventional methods is collaborative filtering which is based on calculating user ratings similarity. In this research we use *recommenderLab* library functions. Good Users (Green Area) are selected as training instances. and again we want to suggest apps to non-good users hoping they install and pay for them. In the first step we train recommendation model using *UBCF* algorithm.

```
# Select Good Users for training
trainIndex <- which(clustRes$cluster == 2)
affinityMatrix<- as(ratings,"realRatingMatrix")
# Train UBCF Recommender Model
recModel<-Recommender(affinityMatrix[trainIndex], method = "UBCF")
```

And then we can predict most likely apps for non-good users using *recModel*. predict will predict ratings for all apps which have not rate for each user. getting for example top 10 of the apps gives a good list to recommend apps to the user.

```
testIndex <- sample(which(clustRes$cluster != 2),10)
# Testing non-good users
recom <- predict(recModel, affinityMatrix[testIndex,], type="ratings")
recomMat <- as(recom,"matrix")
get10 <- function(x){
      sort(x,decreasing = TRUE, index.return = TRUE)$ix[1:10]
}
# Get top-10 recommendations for each test user
res  <- apply(recomMat, MARGIN = 1, FUN = get10)
resTbl <- data.table(userID = testIndex,res)
print(resTbl)
```

```
##     userID V1 V2 V3 V4 V5 V6 V7 V8 V9 V10
##  1:    999 29 46 79 41 30 82 74 13 32  69
##  2:   3287 70 45 41 31 71 68 84 39 14  75
##  3:   5360 66 28 44 70 55 65 91  1 74  83
##  4:   8549 44 54 66 82  4 38 33 78 61  33
##  5:   1617 32  2 50 63 92 10 18 23 17  12
##  6:   8763 23 31 20 84 25  6 39 62 11  20
##  7:   1678 18 75 46 78 73 61 67 45 85  27
##  8:   4584  1  6 48 50  7 58  3 51 57  37
##  9:   4577  6 63 70 38 37 42 65  2 34  30
## 10:   8875 62 47 57 57 84 80 61 29 15  72
```

In the above table each row corresponds to one user and each column is and ID for one of top 10 recommendations for the user.

Summary and Results

In this research we have done some simple analysis and simulated user installations and payments data. After getting and cleaning data we have clustered users to get appropriate recommendation options. Results of this research is listed below:

Looking at Tenure Plot in EDA shows that we have no new user in our system which means either we have all possible users or there is a problem with our advertisment system to get new users!

In average 0.2 of installed apps will get paied for each user.

Including in-app purchases which costs more than 4133 have very little chance to get paied

We have two recommendation systems now: group based and per-user. we can recommend apps using these two systems to users to get paid more hopefully