Big Data Analytics Final Report
Santander Product Recommendation Via Hybrid Approaches

Huilong An, Wenhang Bao, Yifan Zhang
Department of Statistics
Columbia University
e-mail: ha2399@columbia.edu; wb2304@columbia.edu; yz2831@columbia.edu

Abstract
Collaborative filtering(CF) method has been successfully used in recommender systems to support product recommendations but it has some limitations. This work uses both the customer information and customer demands obtained from the frequent purchased products in each category as valuable information. This work designed a hybrid filtering strategy to combines CF and customer's personal information to improve the quality of recommendation. The results prove that the quality of recommendation obtained by this hybrid filtering strategy is outstanding.

Keywords: Hybrid filtering; Collaborative filtering; K-means clustering; Association rules

I. Introduction

With the development of big data technology, many banks began to offer a lending hand to their customers through personalized product recommendations. It is common that we can always receive the bank's mails including some certain product descriptions and recommendations like credit cards invitations. Sometimes we can be interested in the product while sometimes we may just throw the mails.

For this reason, we three are all interested in how the bank's recommendation system works and how to improve the accuracy of recommendations as much as possible.

With a more effective recommendation system in place, commercial banks can better meet the individual needs of all customers and ensure their satisfaction no matter where they are in life.

It is lucky that we found a competition on Kaggle to make a Santander Product Recommendation system and easily downloaded its huge dataset over 2.3 gigabytes we dealt with. In our final project, we explored an effective and efficient recommendation algorithm to predict which bank product a consumer will be most likely to purchase in the following month based on their past behavior and that of similar customers.

Unlike original recommendation algorithms used in homework, we are trying to employ a more advanced model to calculate the similarities between users, which significantly changed the way to customize the recommendation system.

II. Related Works

We searched many existing recommendation algorithms as well as how they work and compare their pros and cons. We have focused on studying content-based filtering, collaborative filtering, weighted RFM-based method and hybrid works.

Content-based filtering techniques like keyword matching recommends items based on a comparison between the content of the items and a user profile. The content of each item is represented as a set of descriptors or terms, typically the words that occur in a document. The user profile is represented with the same terms and built up by analyzing the content of items which have been seen by the user. The idea behind the content-based filtering systems

is that if users liking an item in the past would probably like other similar items in the future. Content-based filtering recommender systems obtain items' characteristics and compare them with users' interest profiles for predicting user preferences.

However, in our dataset we only know if the customer own the certain product but do not really know the characteristics of products. So we do not think it is fit to apply content-based filtering in our project. Then we think about the collaborative filtering approach.

The underlying idea behind collaborative filtering approach is that if a person A has the same opinion as a person B on an issue, A is more likely to have B's opinion on a different issue x than to have the opinion on x of a person chosen randomly. A typical KNN-based collaborative filtering method employs nearest-neighbor algorithm to recommend products to a target customer u based on the preferences of neighbors.

We can construct a customer-item matrix R using customer purchase history, are represented as such that, rij is one if the ith customer purchased the jth product; and is zero otherwise. Then to compute the Pearson correlation.

Customers are ranked by their similarity measures in relation to the target customer u, as determined using the Pearson correlation coefficient. The k most similar (highest ranked) customers are selected as the k-nearest neighbors of customer

u. Finally, the Top-N recommended products are determined from the k-nearest neighbors of u.

From Weighted RFM-based method, we learned how to calculate the similarity among customers based on weighted RFM values of customers. This method gives us an idea how to transform our categorical data including customer personal information like sex into a variable matrix which is able to calculate similarity matrix.

Finally, a hybrid recommender system can help combine different techniques to mutually eliminate their disadvantages. Most hybrid methods applied user profiles and descriptions of items to find users who have similar interests, then used collaborative filtering to make predictions.

III. System Overview

3.1 System Design

3.1.1 Data Preparing and Preprocessing

First of all, we need to clean our dataset including following steps (1) removing outliers and smoothing distribution; (2) filling missing data case by case (3) assigning unknown N/A to empty strings.

During the cleaning of dataset, data visualization including showing the distribution of age and different income levels among different regions also help us explore the characteristics of data in order to clean the data more reasonably.

3.1.2 Constructing Customer-Information Matrix

This method primarily used the structure of weighted RFM-based method. The way how weighted RFM-based method manipulate categorical data is referenced. In this model, the value of Age, Start Date, New Customer Index, Annual Income, etc. are selected as important variables and transformed to customer-information matrix. The similarity among customers is measured by the value of Pearson correlation coefficient based on the normalized weighted information values of customers.

Eq.1

$$Corr_{\text{info}}(C_i, C_j) = \frac{\sum_{s\in V}(W\text{info}_{ci,s} - \overline{W\text{info}_{ci}})(W\text{info}_{cj,s} - \overline{W\text{info}_{cj}})}{\sqrt{\sum_{s\in V}(W\text{info}_{ci,s} - \overline{W\text{info}_{ci}})^2 (W\text{info}_{cj,s} - \overline{W\text{info}_{cj}})^2}}$$

3.1.3 Customer-Demands Matrix

In a real domain, each customer only purchased limited products, making similarity matrix difficult to create due to lack of information. The sparsity of the matrix is a limitation of the collaborative filtering. Moreover, the fact that a customer has not bought a product does not conclude that the customer has no need for it. So the customer demands and past purchased preferences are combined to build the customer demands matrix. The element r_{ij} of the Customer – Demands matrix represents whether the *i*th customer had purchased the *j*th product. If the *i*th customer already purchase the *i*th product, r_{ij} is 1; otherwise it is 0. The similarity of different user can be measured by the value of Pearson correlation coefficient.

Eq.2

$$corr_P(c_i, c_j) = \frac{\sum_{s\in I}(r_{c_i,s} - \bar{r}_{c_i})(r_{c_j,s} - \bar{r}_{c_j})}{\sqrt{\sum_{s\in I}(r_{c_i,s} - \bar{r}_{c_i})^2 \sum_{s\in I}(r_{c_j,s} - \bar{r}_{c_j})^2}}$$

3.1.4 K-means Clustering via Integrated Correlations

Figure 1 illustrates the pattern of the algorithm. The customer information matrix and customer-demands matrix are constructed. Then the correlation coefficients are computed using Pearson correlation coefficient. The K-means clustering is used to cluster customers with similar correlation coefficients. Customer belonging to the same cluster have similar behavior and purchased similar item set. Both of the matrix are normalized. The integrated correlation coefficient is then obtained according to Eq.3. Such a coefficient between the centroid c_j of a cluster and a user c_i is measured using Eq.3. Users are assigned to a cluster with maximum integrated correlation coefficient. The weights of parameters are used to yield an integrated correlation coefficient. After the parameter tuning, W_{info} equals 0.8 and W_{cd} equals 0.2.

Eq.3

$$Corr_{integrated}(C_i, C_j) = W_{info} \times Corr_{info}(C_i, C_j) + W_{cd} \times Corr_{cd}(C_i, C_j)$$

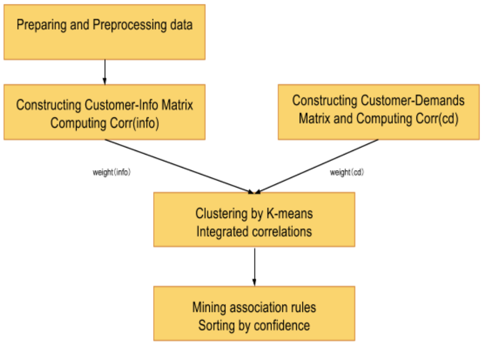
3.1.5 Training association rules

Association rule is used to extract the patterns and to give recommendations. The recommendation engine is based on the transactions associated within each cluster. A cluster is generated by grouping users according to the integrated correlation coefficients. The recommendation engine first identifying the cluster Cj the customer belongs to. Then the set of recommendation rules obtained from cluster C_j is used to select the most recommended product the user didn't purchased. The demands of customer are also used as a factor in recommendation.

3.1.6 Generalized recommendation

Some of the users didn't purchase any product. For these users we could not calculate the Pearson correlation coefficient of the customer demands matrix. For these users, we assigned them to a cluster based only on the customer information matrix and recommend them the most frequently purchased item of that cluster. Some of the users do not have any personal information. For these kind of users, we recommend the most frequently purchased product of the whole dataset. After this phase, the recommendation engine could return a well generalized recommendation for all kind of customers.

Figure 1

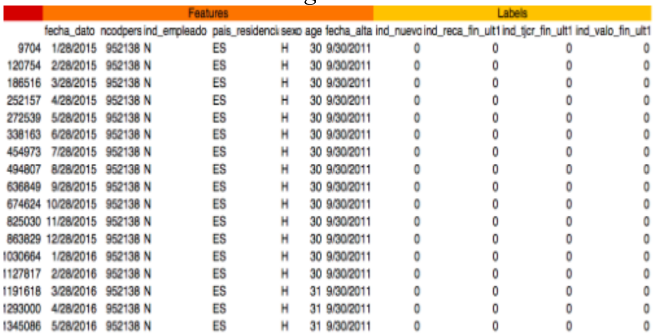


3.2 Datasets

We downloaded our training and test datasets from the following website (https://www.kaggle.com/c/santander-product-recommendation/data) and the uncompressed data size is over 2.3GB.

We are provided with 1.5 years of customers' behavior data from Santander bank to predict what new products customers will purchase. The data starts at 2015-01-28 and has over 10 million monthly records of products a customer has. We can divide the training dataset into 24 feature variables and 24 label variables like figure 2.

Figure 2



	Features							Labels			
	fecha_dato	ncodpers	ind_empleado	pais_residenci	sexo	age	fecha_alta	ind_nuevo	ind_reca_fin_ult1	ind_tjcr_fin_ult1	ind_valo_fin_ult1
9704	1/28/2015	952138	N	ES	H	30	9/30/2011	0	0	0	0
120754	2/28/2015	952138	N	ES	H	30	9/30/2011	0	0	0	0
186516	3/28/2015	952138	N	ES	H	30	9/30/2011	0	0	0	0
252157	4/28/2015	952138	N	ES	H	30	9/30/2011	0	0	0	0
272539	5/28/2015	952138	N	ES	H	30	9/30/2011	0	0	0	0
338163	6/28/2015	952138	N	ES	H	30	9/30/2011	0	0	0	0
454973	7/28/2015	952138	N	ES	H	30	9/30/2011	0	0	0	0
494807	8/28/2015	952138	N	ES	H	30	9/30/2011	0	0	0	0
636849	9/28/2015	952138	N	ES	H	30	9/30/2011	0	0	0	0
674624	10/28/2015	952138	N	ES	H	30	9/30/2011	0	0	0	0
825030	11/28/2015	952138	N	ES	H	30	9/30/2011	0	0	0	0
863829	12/28/2015	952138	N	ES	H	30	9/30/2011	0	0	0	0
1030664	1/28/2016	952138	N	ES	H	30	9/30/2011	0	0	0	0
1127817	2/28/2016	952138	N	ES	H	30	9/30/2011	0	0	0	0
1191618	3/28/2016	952138	N	ES	H	31	9/30/2011	0	0	0	0
1293000	4/28/2016	952138	N	ES	H	31	9/30/2011	0	0	0	0
1345086	5/28/2016	952138	N	ES	H	31	9/30/2011	0	0	0	0

Feature variables includes Age, sex, employment, residence and etc. while label variables are various products a customer has, such as "credit card", "savings account", etc.

They are all dummy variables (0 or 1) to show if the user is currently having the certain product

IV. Algorithm

4.1 Data cleaning and exploration

For the part of data cleaning, we conduct an exploratory data analysis to help it.

4.1.1 Aging Distribution Smoothing

From figure 3, it is obvious that there are some people with very small (below 20) and very high ages (above 100). We thought that these age are not significant for data study because it is low probability for these people to own bank products at this kind of age. For this reason, we separate the original distribution and move the outliers to the mean of the closest one. Then we get a new distribution which is more smoothing like figure 4.

It's also interesting that the distribution is bimodal. There are a large number of university aged students, and then another peak around middle-age, which is really reasonable because both of these two groups have strong work ability as well as income and consumption so that they need bank products to manage their wealth and optimize consumption.

Figure 3

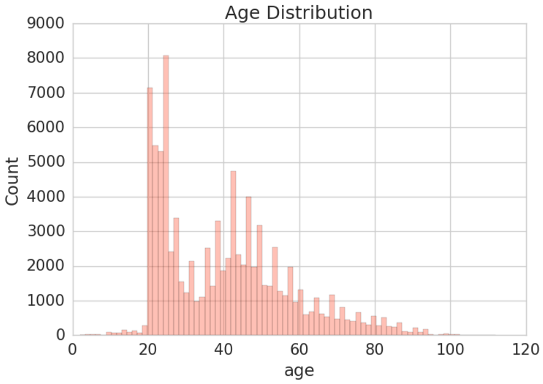
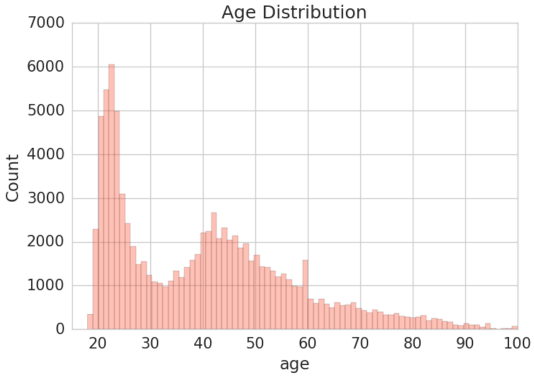


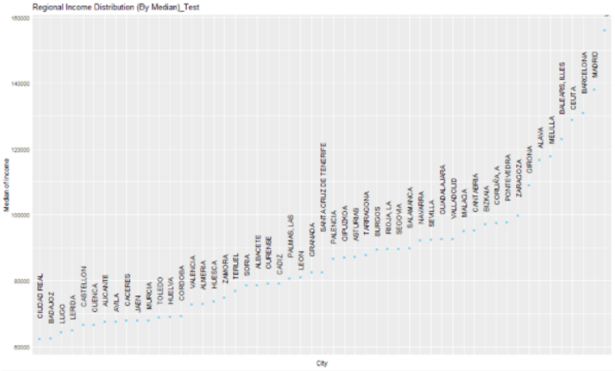
Figure 4



4.1.2 Filling Missing Values

Actually there are some missing values in the original datasets which can make our analysis We fill the missing values by the following three methods: (1) Filling by judging other variables, like the missing values in the variable ind_nuevo, which indicates whether a customer is new or not. We fill in the missing values by judging if the months of history these customers have are over 6 months; (2) Filling by the more common status like indrel; (3) Filling by different medians like what we did in the missing values of incomes. From figure 5, we can see obvious variations between different provinces' medians. So it is not reasonable to apply the total median for all missing values. It is better to assign missing incomes by different medians according to province instead.

Figure 5



4.2 Hybrid Approaches for Recommendation

For the recommendation problem, there are two main challenges here:

The first is how to use personal information. At the beginning, we want to use traditional user-based collaborative filtering, but it can only deal with two-dimensional matrix, and this will definitely lose the personal information. We believe this kind of information is very important, for example, for clients from different age group, the young and old ones might have different purchase behaviors. So we need to do something to capture this kind of information. Here are two ideas we came up with:

1.To use the vector representation about the similarities from personal information and demands information. How to choose the angle of these two vectors actually is a problem. So we discard this kind of method.

2.To use two matrices instead, one named content information matrix, which is used to store personal information, and the demands matrix, which is just like the matrix in user-based CF. Then we calculate similarities from these two matrices and then use the weighted similarities.

$$\text{weightedSimilarity} = \alpha * \text{contentSimilarity} + (1 - \alpha) * \text{demandSimilarity}$$

Then we K-means on the new generated similarity matrix with our own definition. Then we apply association rule to every cluster to make recommendations. This process can capture all the information that is needed.

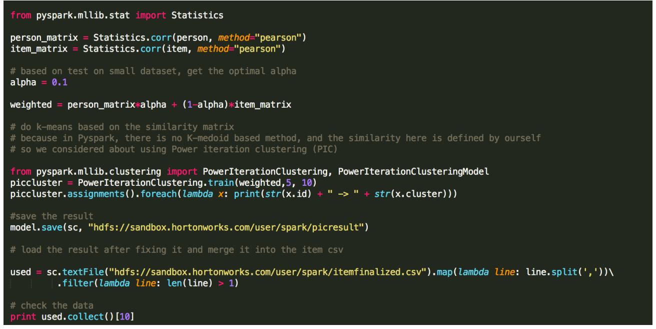
For another problem is how to capture the sequence information, for example, the sequence in which clients brought different products. We did do something to capture this kind of information by making some transformation on the data.

For the tools, we used Python, Spark(pypark), and also R for data cleaning, data analysis and also visualization.

V. SOFTWARE PACKAGE DESCRIPTION

We used pySpark to compute the similarity between customers.

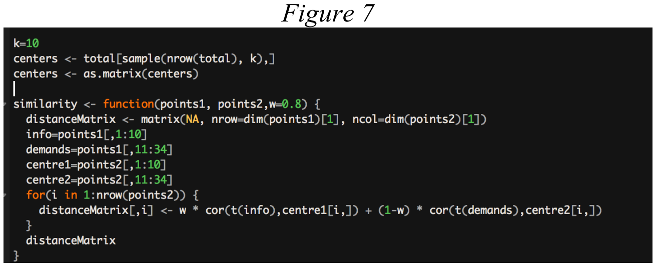
Figure 6



```
from pyspark.mllib.stat import Statistics

person_matrix = Statistics.corr(person, method="pearson")
item_matrix = Statistics.corr(item, method="pearson")

# based on test on small dataset, get the optimal alpha
alpha = 0.1

weighted = person_matrix*alpha + (1-alpha)*item_matrix

# do k-means based on the similarity matrix
# because in Pyspark, there is no K-medoid based method, and the similarity here is defined by ourself
# so we considered about using Power iteration clustering (PIC)

from pyspark.mllib.clustering import PowerIterationClustering, PowerIterationClusteringModel
piccluster = PowerIterationClustering.train(weighted,5, 10)
piccluster.assignments().foreach(lambda x: print(str(x.id) + " -> " + str(x.cluster)))

#save the result
model.save(sc, "hdfs://sandbox.hortonworks.com/user/spark/picresult")

# load the result after fixing it and merge it into the item csv

used = sc.textFile("hdfs://sandbox.hortonworks.com/user/spark/itemfinalized.csv").map(lambda line: line.split(','))\
        .filter(lambda line: len(line) > 1)

# check the data
print used.collect()[10]
```

Then we set the number of clusters with k and set the weight of information matrix.

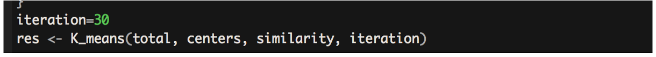
Figure 7



```
k=10
centers <- total[sample(nrow(total), k),]
centers <- as.matrix(centers)

similarity <- function(points1, points2,w=0.8) {
  distanceMatrix <- matrix(NA, nrow=dim(points1)[1], ncol=dim(points2)[1])
  info=points1[,1:10]
  demands=points1[,11:34]
  centre1=points2[,1:10]
  centre2=points2[,11:34]
  for(i in 1:nrow(points2)) {
    distanceMatrix[,i] <- w * cor(t(info),centre1[i,]) + (1-w) * cor(t(demands),centre2[i,])
  }
  distanceMatrix
}
```

After that we set the number of iterations:

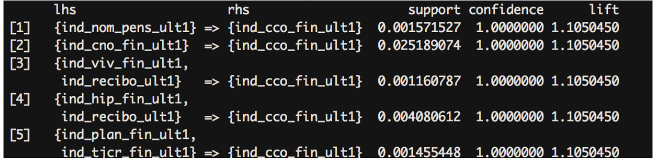
Figure 8



```
}
iteration=30
res <- K_means(total, centers, similarity, iteration)
```

The following figure shows the association rule:

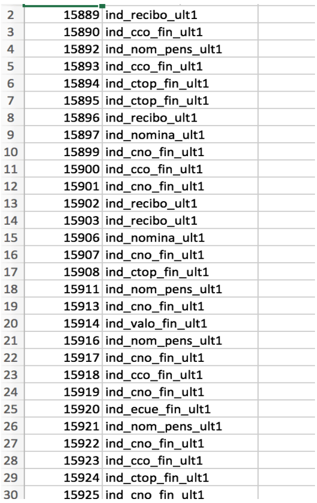
Figure 9



```
      lhs                      rhs                     support confidence      lift
[1]   {ind_nom_pens_ult1} => {ind_cco_fin_ult1}  0.001571527  1.0000000 1.1050450
[2]   {ind_cno_fin_ult1}  => {ind_cco_fin_ult1}  0.025189074  1.0000000 1.1050450
[3]   {ind_viv_fin_ult1,
       ind_recibo_ult1}   => {ind_cco_fin_ult1}  0.001160787  1.0000000 1.1050450
[4]   {ind_hip_fin_ult1,
       ind_recibo_ult1}   => {ind_cco_fin_ult1}  0.004080612  1.0000000 1.1050450
[5]   {ind_plan_fin_ult1,
       ind_tjcr_fin_ult1} => {ind_cco_fin_ult1}  0.001455448  1.0000000 1.1050450
```

Finally we can get our output:

Figure 10



2	15889	ind_recibo_ult1	
3	15890	ind_cco_fin_ult1	
4	15892	ind_nom_pens_ult1	
5	15893	ind_cco_fin_ult1	
6	15894	ind_ctop_fin_ult1	
7	15895	ind_ctop_fin_ult1	
8	15896	ind_recibo_ult1	
9	15897	ind_nomina_ult1	
10	15899	ind_cno_fin_ult1	
11	15900	ind_cco_fin_ult1	
12	15901	ind_cno_fin_ult1	
13	15902	ind_recibo_ult1	
14	15903	ind_recibo_ult1	
15	15906	ind_nomina_ult1	
16	15907	ind_cno_fin_ult1	
17	15908	ind_ctop_fin_ult1	
18	15911	ind_nom_pens_ult1	
19	15913	ind_cno_fin_ult1	
20	15914	ind_valo_fin_ult1	
21	15916	ind_nom_pens_ult1	
22	15917	ind_cno_fin_ult1	
23	15918	ind_cco_fin_ult1	
24	15919	ind_cno_fin_ult1	
25	15920	ind_ecue_fin_ult1	
26	15921	ind_nom_pens_ult1	
27	15922	ind_cno_fin_ult1	
28	15923	ind_cco_fin_ult1	
29	15924	ind_ctop_fin_ult1	
30	15925	ind_cno_fin_ult1	

VI. EXPERIMENT RESULTS

6.1 Description of results

From figure 10, we can see that we used our recommendation system to predict what additional product a customer will get in the last month, 2016-06-28. For example, the customer15889, he would be likely to get ind_recibo_ult1 which means Direct Debit in his following month as our recommendation model outputs.

6.2 Evaluation of the performance

The baseline model in this project utilized the overall item frequency. The recommender sorted the items frequency of the whole dataset and recommend a customer the first item he didn't purchase. This model outcompeted 25% of all competitors. The advanced model described above outcompeted 78% without parameters tuning.

VII. CONCLUSION

7.1 Summary

A successful recommendation engine must have the ability to deal with all kind of customers. The quality of recommendation should be better when the information of the customer be more complete. The recommendation engine need a basic generalized recommendation list to deal with new customers and customers just registered but still purchased nothing.

The design of the system could be complex. But the computation feasibility should be considered into the design of the system.

The hybrid filtering method outcompete collaborative filtering in this situation. It doesn't have limitations like cannot deal with the sparsity of customer demands matrix. Moreover, the content based customer information matrix contains information that is not considered in the traditional collaborative filtering. That is an important attribute for building a customized recommendation engine.

7.2 Group Work Division

An, Huilong:

Material collection and research; Code part of Spark to calculate similarity and k-means clustering; Optimization of algorithm; Result analysis; Report writing

Bao, Wenhang:

Data collection and background research; Design of model; Code part of hybrid recommendation system; Result analysis; Report writing

Zhang, Yifan:

Material collection and reference study; Design of model; Code part of data cleaning, data exploration and visualization; Result analysis; Report writing

ACKNOWLEDGMENT

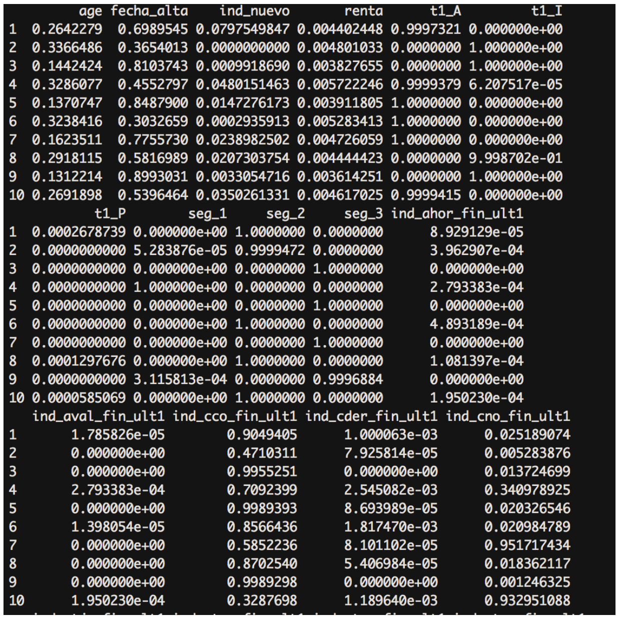
First of all, we would like to show our deepest gratitude to our Professor Ching-Yung Lin, a responsible and resourceful scholar, who provided us an opportunity to know about big data technology such a fast changing field. Novelty he taught us is also important to our continuous study especially in this competitive environment.

We also shall extend our many thanks to TAs for all their kindness and help in our study, especially many tutorials given by Eric.

Last but not least, we would like to thank classmates we meet in the class, for their peer encouragement.

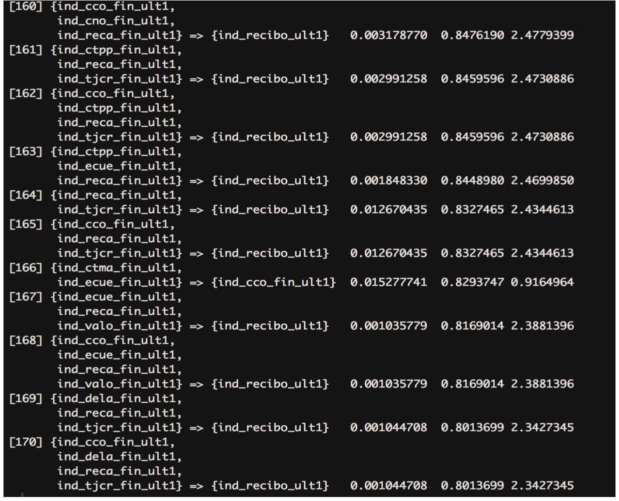
APPENDIX

Centers of each cluster:



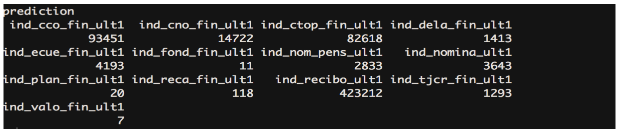
```
          age fecha_alta    ind_nuevo       renta      t1_A         t1_I
1  0.2642279  0.6989545 0.0797549847 0.004402448 0.9997321 0.000000e+00
2  0.3366486  0.3654013 0.0000000000 0.004801033 0.0000000 1.000000e+00
3  0.1442424  0.8103743 0.0009918690 0.003827655 0.0000000 1.000000e+00
4  0.3286077  0.4552797 0.0480151463 0.005722246 0.9999379 6.207517e-05
5  0.1370747  0.8487900 0.0147276173 0.003911805 1.0000000 0.000000e+00
6  0.3238416  0.3032659 0.0002935913 0.005283413 1.0000000 0.000000e+00
7  0.1623511  0.7755730 0.0238982502 0.004726059 1.0000000 0.000000e+00
8  0.2918115  0.5816989 0.0207303754 0.004444423 0.0000000 9.998702e-01
9  0.1312214  0.8993031 0.0033054716 0.003614251 0.0000000 1.000000e+00
10 0.2691898  0.5396464 0.0350261331 0.004617025 0.9999415 0.000000e+00
           t1_P        seg_1     seg_2     seg_3 ind_ahor_fin_ult1
1  0.0002678739 0.000000e+00 1.0000000 0.0000000      8.929129e-05
2  0.0000000000 5.283876e-05 0.9999472 0.0000000      3.962907e-04
3  0.0000000000 0.000000e+00 0.0000000 1.0000000      0.000000e+00
4  0.0000000000 1.000000e+00 0.0000000 0.0000000      2.793383e-04
5  0.0000000000 0.000000e+00 0.0000000 1.0000000      0.000000e+00
6  0.0000000000 0.000000e+00 1.0000000 0.0000000      4.893189e-04
7  0.0000000000 0.000000e+00 0.0000000 1.0000000      0.000000e+00
8  0.0001297676 0.000000e+00 1.0000000 0.0000000      1.081397e-04
9  0.0000000000 3.115813e-04 0.0000000 0.9996884      0.000000e+00
10 0.0000585069 0.000000e+00 1.0000000 0.0000000      1.950230e-04
   ind_aval_fin_ult1 ind_cco_fin_ult1 ind_cder_fin_ult1 ind_cno_fin_ult1
1       1.785826e-05        0.9049405      1.000063e-03      0.025189074
2       0.000000e+00        0.4710311      7.925814e-05      0.005283876
3       0.000000e+00        0.9955251      0.000000e+00      0.013724699
4       2.793383e-04        0.7092399      2.545082e-03      0.340978925
5       0.000000e+00        0.9989393      8.693989e-05      0.020326546
6       1.398054e-05        0.8566436      1.817470e-03      0.020984789
7       0.000000e+00        0.5852236      8.101102e-05      0.951717434
8       0.000000e+00        0.8702540      5.406984e-05      0.018362117
9       0.000000e+00        0.9989298      0.000000e+00      0.001246325
10      1.950230e-04        0.3287698      1.189640e-03      0.932951088
```

The sample rules within each cluster:



```
[160] {ind_cco_fin_ult1,
       ind_cno_fin_ult1,
       ind_reca_fin_ult1} => {ind_recibo_ult1}   0.003178770  0.8476190 2.4779399
[161] {ind_ctpp_fin_ult1,
       ind_reca_fin_ult1,
       ind_tjcr_fin_ult1} => {ind_recibo_ult1}   0.002991258  0.8459596 2.4730886
[162] {ind_cco_fin_ult1,
       ind_ctpp_fin_ult1,
       ind_reca_fin_ult1,
       ind_tjcr_fin_ult1} => {ind_recibo_ult1}   0.002991258  0.8459596 2.4730886
[163] {ind_ctpp_fin_ult1,
       ind_ecue_fin_ult1,
       ind_reca_fin_ult1} => {ind_recibo_ult1}   0.001848330  0.8448980 2.4699850
[164] {ind_reca_fin_ult1,
       ind_tjcr_fin_ult1} => {ind_recibo_ult1}   0.012670435  0.8327465 2.4344613
[165] {ind_cco_fin_ult1,
       ind_reca_fin_ult1,
       ind_tjcr_fin_ult1} => {ind_recibo_ult1}   0.012670435  0.8327465 2.4344613
[166] {ind_ctma_fin_ult1,
       ind_ecue_fin_ult1} => {ind_cco_fin_ult1}  0.015277741  0.8293747 0.9164964
[167] {ind_ecue_fin_ult1,
       ind_reca_fin_ult1,
       ind_valo_fin_ult1} => {ind_recibo_ult1}   0.001035779  0.8169014 2.3881396
[168] {ind_cco_fin_ult1,
       ind_ecue_fin_ult1,
       ind_reca_fin_ult1,
       ind_valo_fin_ult1} => {ind_recibo_ult1}   0.001035779  0.8169014 2.3881396
[169] {ind_dela_fin_ult1,
       ind_reca_fin_ult1,
       ind_tjcr_fin_ult1} => {ind_recibo_ult1}   0.001044708  0.8013699 2.3427345
[170] {ind_cco_fin_ult1,
       ind_dela_fin_ult1,
       ind_reca_fin_ult1,
       ind_tjcr_fin_ult1} => {ind_recibo_ult1}   0.001044708  0.8013699 2.3427345
```

Predictions table:



```
prediction
 ind_cco_fin_ult1  ind_cno_fin_ult1 ind_ctop_fin_ult1 ind_dela_fin_ult1
            93451             14722             82618              1413
ind_ecue_fin_ult1 ind_fond_fin_ult1 ind_nom_pens_ult1   ind_nomina_ult1
             4193                11              2833              3643
ind_plan_fin_ult1 ind_reca_fin_ult1   ind_recibo_ult1 ind_tjcr_fin_ult1
               20               118            423212              1293
ind_valo_fin_ult1
                7
```

REFERENCES

[1] Ya –Yueh Shih and Duen-Ren Liu, “Hybrid recommendation approaches: collaborative filtering via valuable content information” 2015 IEEE

[2] Linden, G., Smith, B., and York, J., “Amazon.com recommendations: item-to-item collaborative filtering. Internet”, Computing IEEE, 7(1), 2003, pp. 76-80.

[3] https://www.kaggle.com/apryor6/santander-product-recommendation/detailed-cleaning-visualization-python

[4] Claypool, M., Gokhale, A., and Miranda, T., “Combining content-based and collaborative filters in an online newspaper”, Proceedings of the ACM SIGIR Workshop on Recommender Systems, Berkeley, 1999.

[5] Toine Bogers, Antal van den Bosch, “Collaborative and Content-based Filtering for item Recommendation on Social Bookmarking Websites” Unpublished

[6] Salton, G., and McGill, M.J., “Introduction to Modern Information Retrieval”, McGraw-Hill, New York, 1983.

[7] Basu, C., Hirsh, H., and Cohen, W., “Recommendation as classification: using social and content-based information in recommendation”, Proceedings of the Fifteenth National Conference on Artificial Intelligence, Madison, 1998, pp. 714-720.

[8] Kim, J.W., Lee, B.H., Shaw, M.J., Chang, H.L., and Nelson, M., “Application of decision-tree induction techniques to personalized advertisements on Internet storefronts”, International Journal of Electronic Commerce, 5(3), 2001, pp. 45-62.

[9] Liu, D.R., and Shih, Y.Y., “Integrating AHP and data mining for product recommendation based on customer lifetime value”, Information & Management (Accepted), 2004.

[10] Li, Q., and Kim, B.M., “An approach for combining content-based and collaborative filters”