Santander Product Recommendation System

Junjie Huang
CIMS
New York University
jh5501@nyu.edu

Lu Yin
Center for Data Science
New York University
ly1123@nyu.edu

Shenghui Zhou
Center for Data Science
New York University
sz2396@nyu.edu

Abstract

In this project, we aims to implement well structured recommendation systems that help Santander Bank to provide accurate product recommendations based on their 1.5 years customers purchase records. We choose Simple Recommender, User-Based Recommender and Random Forest multi-classification model for all customers, then we apply XGboost on target customers (who tend to buy new products) as our improvement after doing data filtration. We choose MAP (mean average precision) and hit rate as the main metrics to measure these models. As a result, we compare the performance and drawbacks of each model based on different purposes of recommendation.

1 Introduction and motivation

In the contemporary society, e-commerce has significantly changed people's lives with the development of Internet and digital technologies, most of the banks are trying to provide a list of recommendations to their customers with the purpose of increasing sales and retaining customers. However, random suggestion will cause low chance for customers to buy new products. Hence, implementing a well structured recommendation system is becoming challenging and important for every bank.

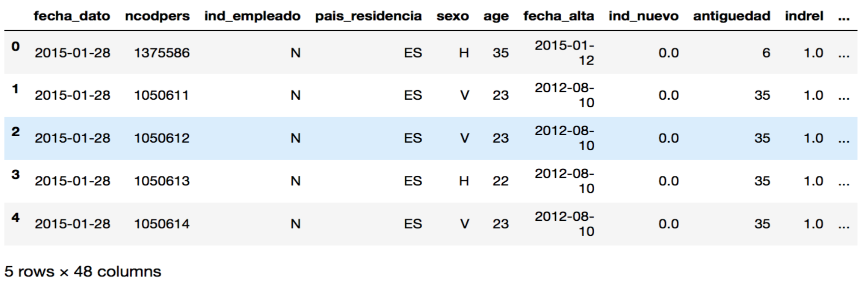
In 2017, Santander Bank, who previously had recommendation system with low accuracy and few records, published a challenge on Kaggle Competition that aims to predict product purchase for each customer in following month based on the historical purchase records. The dataset provides the monthly records of customer data with 1.5 years period ranging from 1/28/2015 to 5/28/2016 with 48 features both on demographics and product purchase records.

In this project, our group aims to provide well structured recommendation systems that provide sorted top 7 products that the customers are likely to buy in the following month exclude the products they already bought previously. We use both traditional recommender(popularity and similarity) and multi-classification model(Random Forest for our recommendation system) to make predictions. We will provide detailed procedures and comparisons on developing these recommendation systems in following parts of our report.

2 Data cleaning and exploration

2.1 Data Description

This dataset provides the monthly records of customer behavior data with 1.5 years period ranging from 1/28/2015 to 5/28/2016 including 24 demographic features such as 'Age','Sex', 'Customer code' and 24 product features which are the names of the products like 'Guarantees','Saving Account' and 'Credit Card' etc. We have a list of demographic information and the products owned for each individual in the period of the one and half years. The table of dataset description has been shown below:



	fecha_dato	ncodpers	ind_empleado	pais_residencia	sexo	age	fecha_alta	ind_nuevo	antiguedad	indrel	...
0	2015-01-28	1375586	N	ES	H	35	2015-01-12	0.0	6	1.0	...
1	2015-01-28	1050611	N	ES	V	23	2012-08-10	0.0	35	1.0	...
2	2015-01-28	1050612	N	ES	V	23	2012-08-10	0.0	35	1.0	...
3	2015-01-28	1050613	N	ES	H	22	2012-08-10	0.0	35	1.0	...
4	2015-01-28	1050614	N	ES	V	23	2012-08-10	0.0	35	1.0	...

5 rows × 48 columns

However, there are several limitations exist in this dataset:

1. The huge dataset including over 13.6 millions of records, which might cause difficulties on modeling in terms of expensive computation and long training time.
2. There are about 20 percent of missing data on demographic feature, which need to be replaced with estimations focusing on numerical features or categorical features.
3. There are several records for the same customer with different time periods.
4. Product information are vague with only product name and indicator of 0 and 1 but not the true amount of product purchased.

2.2 Data Cleaning, preprocessing and some visualization

In this section, we aim to provide a brief understanding of the dataset, clean and pre-process the dataset feature by feature and gives visualization of the numerical features.

2.2.1 Categorical Features

For each column of the categorical feature, we firstly fill in all the missing value as 'nan' type, then we list all the unique values of each feature in order to give a more accurate classification of each column. Some of the examples are shown below:

This table shows the nation records of each customer, we can find that there is no strange values, hence we keep all the categories.

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

This table shows the raw unique value for customer type, we can find that some of the value are actually the same, hence, we combine the values into 5 categories from 1 to 4 and 'nan'.

```
Unique values for indrel_1mes:
[1.0 3.0 nan 2.0 '1' '1.0' '3' '3.0' 'P' '2' '4' '2.0' '4.0' 4.0]
```

2.2.2 Numerical Features

1. **Age**: There are a lot of missing values in this column. In addition to null values, it also shows records of people with extremely high and low ages. We take the people between

18 to 100 as our target customers, take people with more than 100 years old as outliers and replace these value with mean of the age. After outliers cleaning, plot the distribution of customers' age as shown in the figure 1 below:

From this figure we can find that lots of people are university aged students and the customers with middle-age is also high.

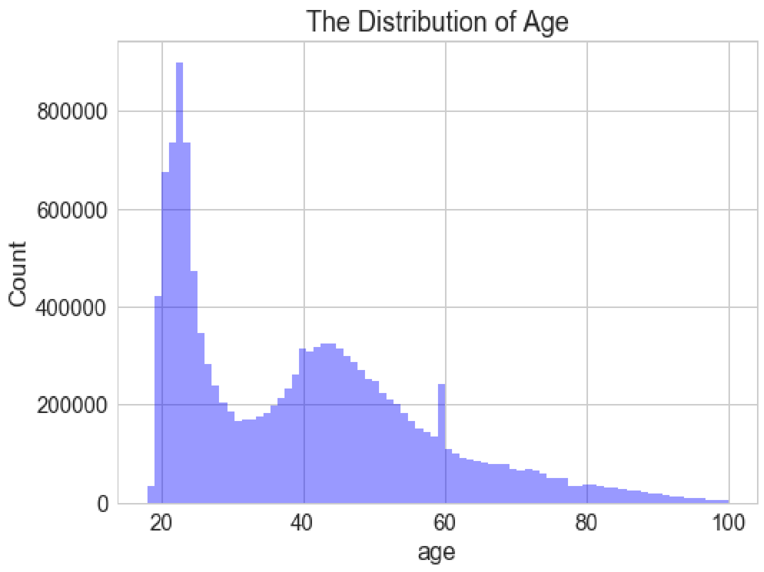


Figure 1: Distribution of Customer Age

2. **Net Income**: This attribute gives the net income for each customer corresponding to every purchase record. Firstly, we find that there are lots of missing values, we fill in the missing value with the mean value of the same location. If the location is also a missing value, we fill in the mean value of the income among all the 'unknown' records. The plot for each city has been plotted in figure 2:

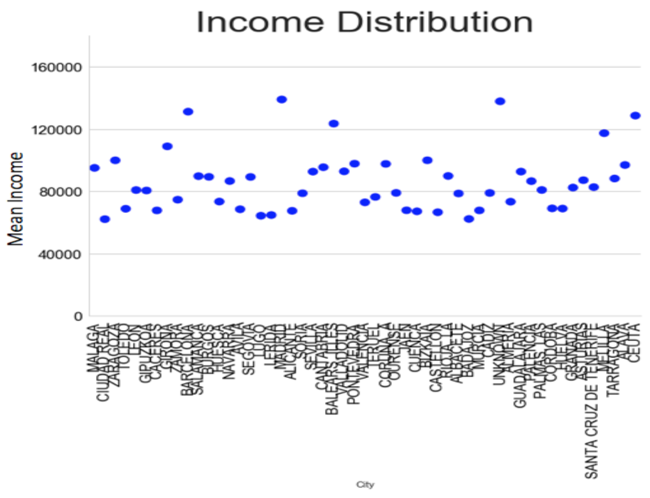


Figure 2: Distribution of Customer Income

2.3 Evaluation Metrics

In the evaluation part of our project, we use MAP (Mean Average Precision) and hit rate to evaluate the performance of our model.

1. **MAP**: the equation of MAPE for n items recommender is listed as follow:

$$MAP@N = \frac{1}{|U|}\Sigma U|(AP@N)_u = \frac{1}{|U|}\Sigma^{|}_{u=1}U|\frac{1}{m}\Sigma^{|}_{u=1}U|\frac{1}{m}\Sigma^{N}_{k=1}P_u(k)rel_u(k) \quad (1)$$

   where N denotes the number of items that we want to recommend, $|U|$ denotes the number of customers, $rel(k)$ is an indicator that says whether that k_{th} item was relevant $(rel(k) = 1)$ or not $(rel(k) = 0)$, $AP@N$ is the average precision for n items recommender, which formulated as: $AP@N = \frac{1}{m,N}\Sigma_{k=1}^{N}P_u(k)rel(k)$.

2. **Hit Rate**: the Hit Rate that we use in this project is defined as "successful recommendation rate". If our recommendations catch the purchase next month, then the hit count for a certain user is 1. We divide the total counts of successful catch by the total number of recommendations we made to get the hit rate percentage.

   In programming language this calculation is expressed as:

```
for a certain user id:
        If the set of predict items ∩ set of test items is not null:
            hit count for this user id = 1
```

Compared with MAP, Hit Rate matters more on whether the recommendation successfully catch the taste of a customer or not. If we increase the amount of recommendations based on score functions, we could see a decrease in average precision but on the other hand, Hit Rate would be increased because of the expanding recommendations. Therefore, the model decision is made based on different purposes of recommendation.

3 Traditional recommender system: popularity, similarity models

In this section, we conduct popularity and user based similarity models as the traditional recommender system.

To gather as much information as we could and calculate tf-idf for similarity recommender, the test set is the data in the last month (5/28/2016) of the data set. And the rest of the data points is split into training set. (1/28/2015 - 4/28/2016)

3.1 Simple Recommender

This is also called the “popularity recommender” for which a weighted rating system was used to calculate the number of products buying on which a descending sorting is finally performed and the top 7 products are recommended. The public choice is sometimes very powerful for developing recommender system.

The steps of the this popularity algorithm are as follows:

1. For all products from the users’ record, count the products with value = 1, which indicates the user buy the product
2. Sort in descending order of all count values associated with product name
3. Pick the top 7 products with highest counts
4. Recommend the same 7 products to all users based on unique user id

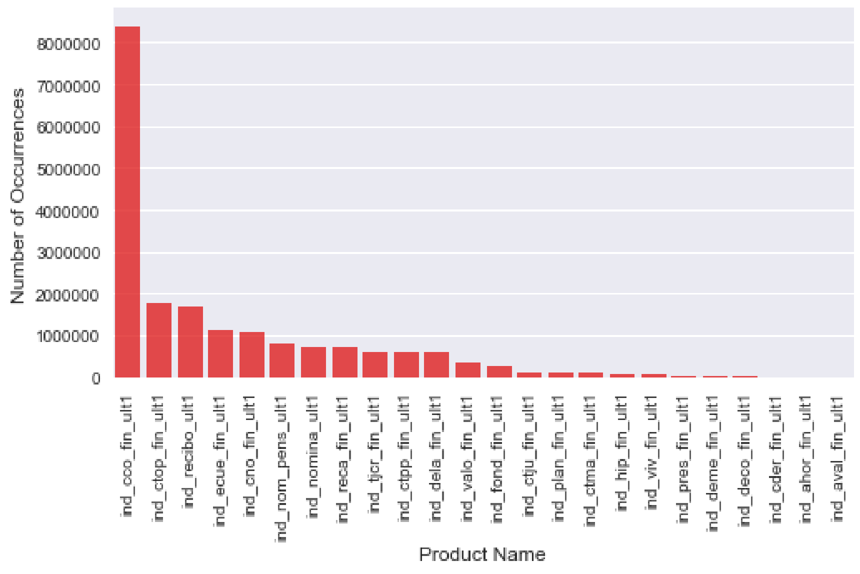
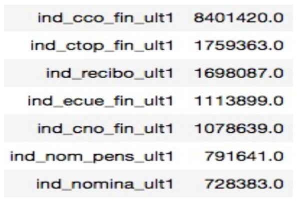


Figure 3: The sorted frequency of products

As it shows on the graph, the popularity recommender is going to recommend the top 7 products which are:



ind_cco_fin_ult1	8401420.0
ind_ctop_fin_ult1	1759363.0
ind_recibo_ult1	1698087.0
ind_ecue_fin_ult1	1113899.0
ind_cno_fin_ult1	1078639.0
ind_nom_pens_ult1	791641.0
ind_nomina_ult1	728383.0

3.2 User-Based Recommender

The cosine similarity is used here to measure the user-user similarity. However, we do not directly calculate the similarity score. Based on the times that a certain product appears for a user, our team use the similar frequency-inverse document frequency (tf-idf) statistic to indicate how important a product is to a certain user. For example, if a certain product appears at almost every user's account, it has a lower priority to be recommended when calculating similarity scores.

The steps of this user-based recommender are as follows:

1. For all products from the users' record, count the products with value = 1, which indicates the user buy the product
2. Calculate TF-IDF statistics based on count values for each unique user

   For a certain user:

   tf (term frequency) in this project is the number of times that a certain product t appears in a user's account (u), where tf(t,u) = 1 if items t appears in user's account and 0 otherwise.

   idf (inverse document frequency) in this project is calculated by the logarithm of quotient that divides the total number of unique users by the number of users containing the certain product.

$$idf(t, U) = \log \frac{N}{|\{u \in U : t \in u\}|} \tag{2}$$

   where N is total number of unique users, U is the users group, u represents each user and t denotes as the product in user's account

Then, tf-idf is calculated as the product of tf and idf:

$$tf - idf = tf * idf \tag{3}$$

The following two plots show the comparison of tf and idf, we can find that the significant of the product that everyone purchase is reduced, which gives better recommendation result since the suggestion for the product that everyone purchases is useless.

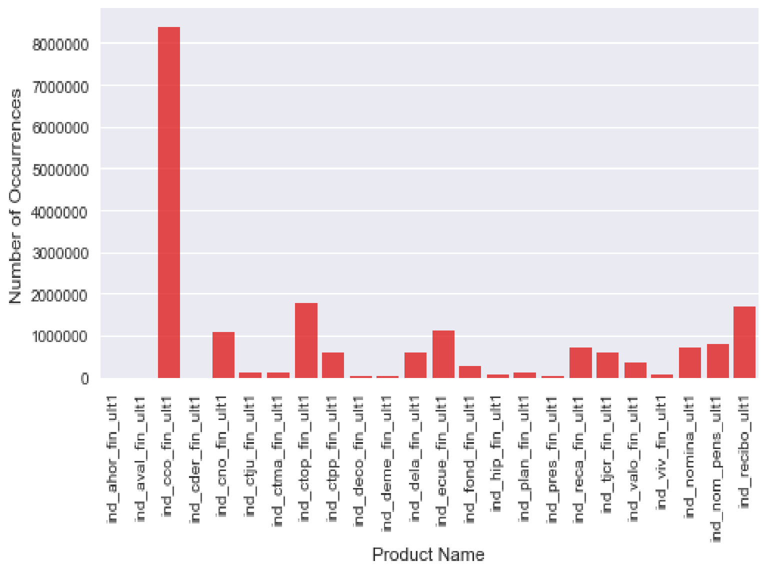


Figure 4: Figure of the number of occurrence for each product (tf)

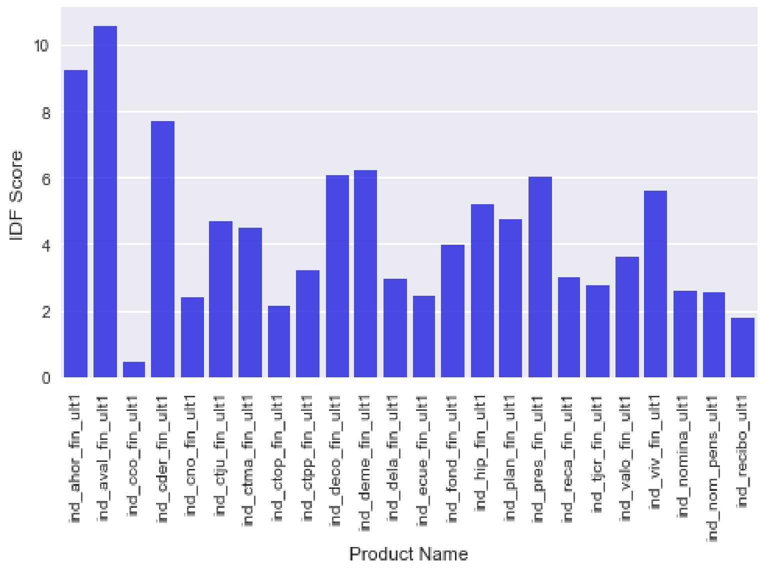


Figure 5: Figure of inverse document frequency for each product (idf)

3. Then based on new tf-idf score, we calculate the pairwise cosine similarity for each user:

$$Similarity = \cos(\theta) = \frac{A\dot{B}}{||A||\dot{|}|B||} = \frac{\Sigma_{i=1}^{n} A_i B_i}{\sqrt{\Sigma_{i=1}^{n} A_i^2}\sqrt{\Sigma_{i=1}^{n} B_i^2}} \tag{4}$$

4. Sort and output top similar users for each user id based on cosine similarity score.
5. Search through this top user lists and recommend products that the user didn't have last month.

For experiment, the top similar users are chosen from the range of 50 to 500. As the result shows, 50 similar users perform best on to maximize MAP score. On the contrary, the hit rate increases as the amount of similar uses increase. Based on different purposes of recommendation system, system with higher top similar users has less MAP score, but have a higher score on Hit Rate.

The MAP of our primary result is really high around 80% but when we look at the hit rate, it appears around 0.02. The reason of this interesting result is because most of the input (recommendation) and output(test) are null values, which means no recommendations are actually made after removing previous purchases. The precision with both null values in prediction and test is primarily set as 1.

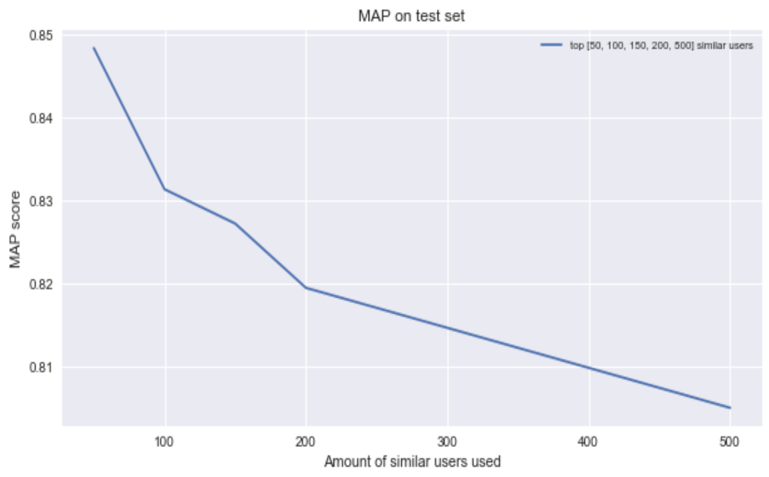


Figure 6: MAP before customer filter

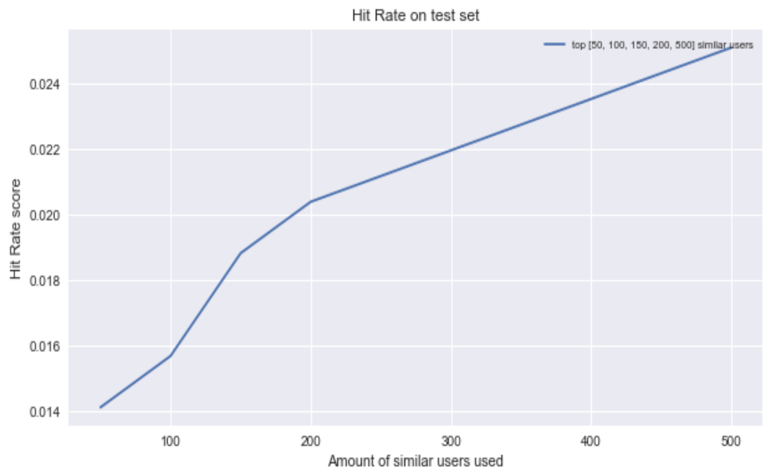


Figure 7: Hit Rate before customer filter

It is intuitive that customers, especially bank customers are too lazy to try new products. As our data results shows, only no more than 5% of the customers are actually try 1 or 2 new financial products next month. Therefore, for our next step, a more advanced filter of customers is performed when calculating MAP and Hit Rate.

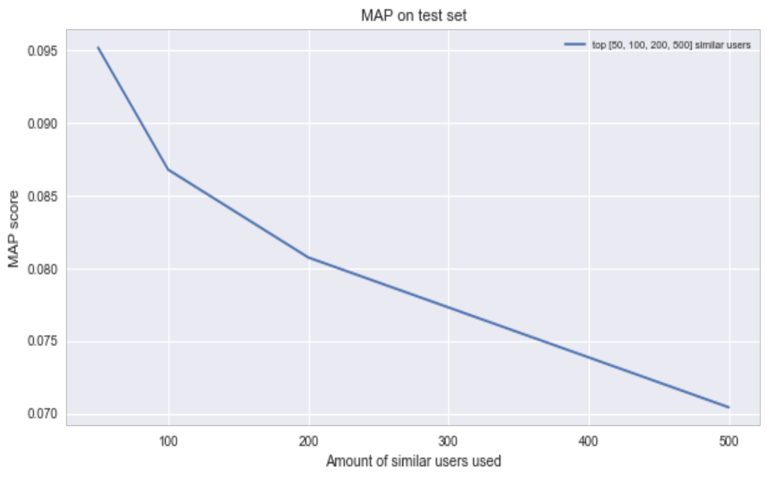


Figure 8: MAP after customer filter

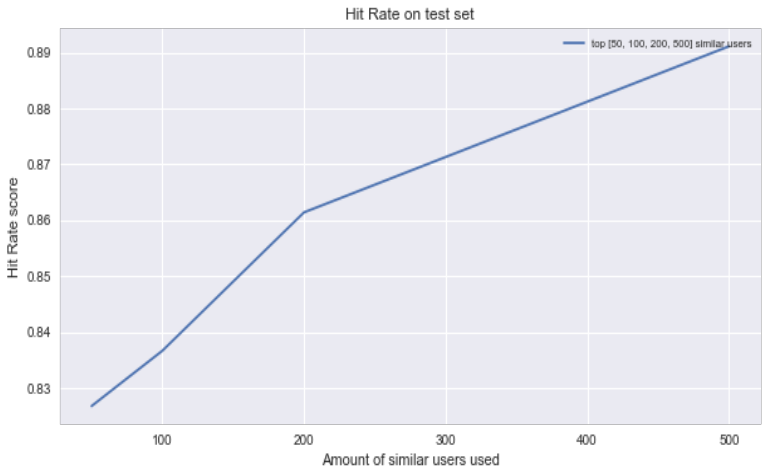


Figure 9: Hit Rate after customer filter

Compare with average MAP score of around 0.03 on Kaggle competition, our prediction MAP result with the first 10000 user id data works well with a value around 0.085. And the hit rate among target customers is around 85%. However, none of the demographic information is used in traditional recommender. It is possible to improve our user similarity based on more demographic information, but it is hard to know the weight between each attributes and how to implement them with purchase behaviors. Therefore, we are going to try classification models that are discussed in next session.

Another challenge of similarity recommender system is that the pairwise similarity takes a really long running time to calculate for large dataset. In this project the process time of the similarity calculation is close to $O(n^2)$. That's the reason why we sample the first 10000 users for our training set. Therefore, running time could another attribute when choosing models.

To make decisions based on MAP and Hit Rate is not easy since business purpose and goal of the recommender could be different. In addition, the difference between off-line evaluation (MAP) and on-line evaluation such as A/B test in terms of business values is discussed in last session of this paper.

4 Multi-classification model: Random Forest model and XGBoost

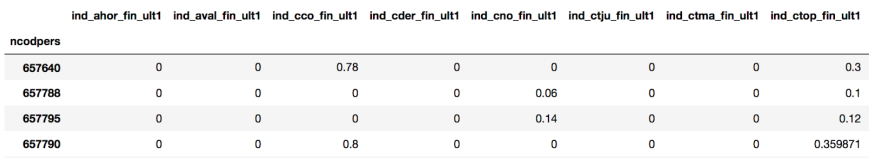
4.1 Baseline: Random Forest

One of the obvious drawbacks of similarity model is that we do not use the demographic information at all. And those demographic information could be really useful when predicting future purchase. Therefore, in this section, we want to apply the classic Machine Learning model that takes in customer information and solves this as a multi-classification problem. We choose Random Forest as our baseline model to start with.

Since we split the data in May 2016 as our validation part from the whole dataset, we must select the appropriate the training data. Considering the seasonal effect, it is reasonable that we use the data in May 2015 as our training data to eliminate possible seasonal effect. For example, in September, the college account will be recommended more than any other products in the bank. So we think 12 months interval is a good choice.

Then we try to build a Random Forest model for our dataset. Since it is a first look at our dataset, we set the number of trees as 50, the number of the maximum futures as square root and the criterion as entropy.

After running Random Forest, we get different probabilities for multiple labels like the below.

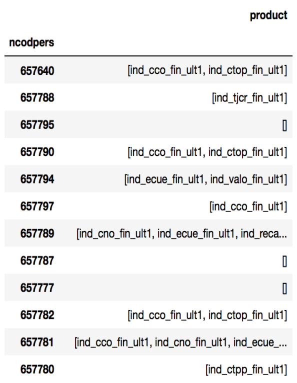


	ind_ahor_fin_ult1	ind_aval_fin_ult1	ind_cco_fin_ult1	ind_cder_fin_ult1	ind_cno_fin_ult1	ind_ctju_fin_ult1	ind_ctma_fin_ult1	ind_ctop_fin_ult1
ncodpers								
657640	0	0	0.78	0	0	0	0	0.3
657788	0	0	0	0	0.06	0	0	0.1
657795	0	0	0	0	0.14	0	0	0.12
657790	0	0	0.8	0	0	0	0	0.359871

For specific user, we sort the product by the probability of each item and output our recommendation with a certain emission threshold. The strategy that we select product is following:

1. Filter the product whose probability is larger than the threshold that we set
2. Choose 7 products which have the highest probability. (If the amount of products are less than 7, just take them all)

Then we set the threshold as 0.3 in this section to see a basic result, we will explore more with the relationship between threshold and MAP in the next section. Here is our results:



	product
ncodpers	
657640	[ind_cco_fin_ult1, ind_ctop_fin_ult1]
657788	[ind_tjcr_fin_ult1]
657795	[]
657790	[ind_cco_fin_ult1, ind_ctop_fin_ult1]
657794	[ind_ecue_fin_ult1, ind_valo_fin_ult1]
657797	[ind_cco_fin_ult1]
657789	[ind_cno_fin_ult1, ind_ecue_fin_ult1, ind_reca...
657787	[]
657777	[]
657782	[ind_cco_fin_ult1, ind_ctop_fin_ult1]
657781	[ind_cco_fin_ult1, ind_cno_fin_ult1, ind_ecue_...
657780	[ind_ctpp_fin_ult1]

We get the MAP as 0.006049 and Hit Rate is 0.036638. The result looks not so good. We need do some improvement in our models.

One of the issues we have is that we train all the users in May 2015 rather than users who buy the new product in this month. It could be misleading and reduce our accuracy since most of the customer do not actually buy new products compared with our recommendations. We notice that for classification model, our training dataset may be too broad to make sound predictions. So we try to filter the users who buy products in the next step to see whether there is some improvement for our model.

4.2 Exploration of Machine Learning Method with a preciser filter of target users:

4.2.1 Why we make this improvement on training group

Since the purpose of our project is to predict which products will be added in the next month, and we found that only few of users have buy the new products in every month. We took first 1000 users in 2015-05-28, there are only 14 people which buy the new products in that month. It means that most of the data and information is invalid. The real train dataset is set that has user information who buyt the new products based on the last month items' record. So we should filter our dataset to get a valid training data for our prediction.

4.2.2 Algorithm of user filter

Since we ignored the seasonal effect at the beginning, we found that the distribution of products added may vary every month. For example, the college account is more added than anytime else in the begin of the new seme semester. Since our dataset is 1.5 years behaviour record of Santander Bank. There are more than one billion records in our dataset, the work of filtration is very heavy. So for the improved recommender, we choose the user records of 05/28/2016 as validation data. Then we use the dataset of 04/28/2015 and 05/28/2015 to select the user who buy the new product in the 05/28/2015 as our training set. This is the valid training set we actually want.

The filtration steps are following:

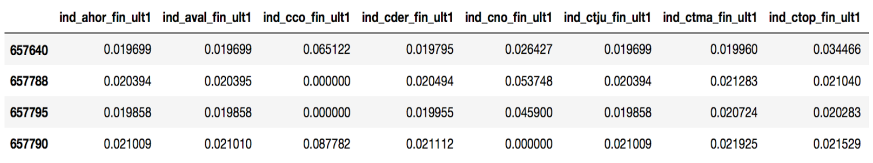
1. Select the common user in the 04-28-2015 and 05-28-2015, (find the product difference of the specific user) then we make the product of 04-28-2015 subtracted by the the product of 05-28-2015. There are three conditions for the products. 1 means the product added, 0 means the product unchanged and -1 represents the product cancelled.
2. Replace -1 by 0 for the simplicity, which means no items cancelled.
3. Sum up the product difference of the specific user. If the sum is larger than 0, this means that this user buy new product in this month. So we add them to a new dataset. Otherwise, the user didn't but the new product. For the recommendation system, this record whose sum is 0 is ignored by us for the training data.

After this, we get a new dataset of all users who buy new products in May 2016.

For using XGBoost in our new dataset after filtration, we need to convert the multiple binary dependent variable into a single multi-value dependent variable. If an user added different products in 05-28-2015, we duplicate the line of the user data in our training data multiple times. For every line, we assign different values to the different added product. In the end, we consider this problem as a single label multi-classification problem.

4.2.3 XGBoost model

XGBoost model is the most popular model in the competition nowadays. It combines the gradient boost methods and ensemble methods. In regular, its results will be better than the other machine learning models such as Random Forest, Logistic Regression and so on. Since our problem is a single label multi-classification problem, we choose the multi: softprob as our objective function. Then we can get the probability for every added production.



	ind_ahor_fin_ult1	ind_aval_fin_ult1	ind_cco_fin_ult1	ind_cder_fin_ult1	ind_cno_fin_ult1	ind_ctju_fin_ult1	ind_ctma_fin_ult1	ind_ctop_fin_ult1
657640	0.019699	0.019699	0.065122	0.019795	0.026427	0.019699	0.019960	0.034466
657788	0.020394	0.020395	0.000000	0.020494	0.053748	0.020394	0.021283	0.021040
657795	0.019858	0.019858	0.000000	0.019955	0.045900	0.019858	0.020724	0.020283
657790	0.021009	0.021010	0.087782	0.021112	0.000000	0.021009	0.021925	0.021529

As we do in the baseline model, we need to selected the products which has highest probability excluding the product the user have already bought.

The steps is following:

1. For all users in the test data, we compare it with the user record in 04-28-2016. If the product indicator in April is 1, which means this user already have bought this product, we set the product probability by 0.
2. Sort the product by the probability.
3. Set a threshold, we filter the products whose probability is larger than the threshold.
4. Select 7 products which have the highest probability. (If the amount of products are less than 7, just take them all).

Then we calculate the MAP and hit rate of our prediction by different threshold we set. Since we know there are many empty recommendation in our prediction results. At first, we set the precision as 1 when we have empty recommendation for the user who don't buy the new product. The figure and figure are the results of MAP and hit rate versus threshold. As we see in the figure, when the threshold is larger than 0.30, the MAP is close to 1. However, the hit rate is almost 0. This is a strange phenomenon. After we check the prediction results, we find that there are only 15 users who have product recommended among 10000 test users since we set the threshold too higher. And among 10000 test users there are only 265 users who have new product added. Hence, there are nearly 250 users who get the wrong recommendation. What's more, there are only 0.15% users who get the recommendation. But we have total 10000 test users. So the MAP is nearly 0.975. However, the hit rate is almost 0. The main issue is that there are 97% of our test users who have no product added. Thus we try to eliminate these users and get another results.

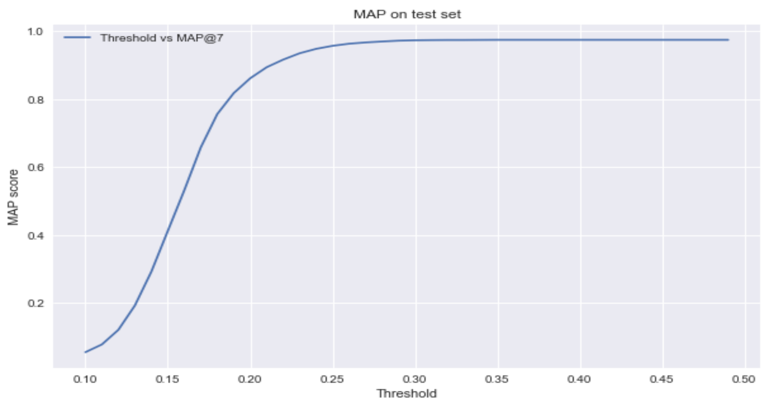


Figure 10: MAP before customer filter

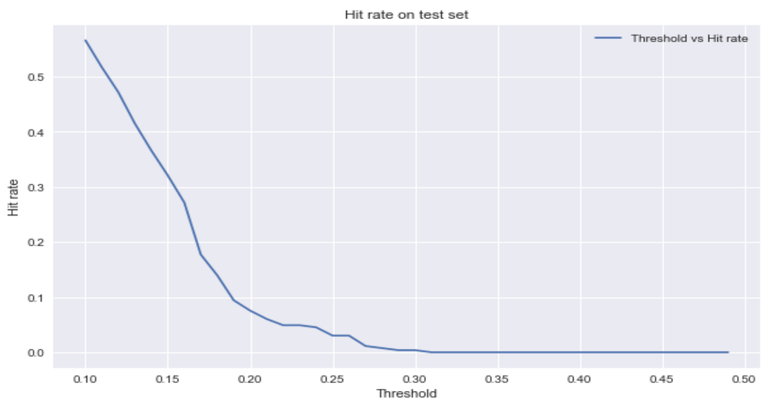


Figure 11: Hit Rate before customer filter

Here are the results after we eliminate those users who don't buy any new products in May 2016. These two plots look more reasonable than the above two. We want to compare this result with the user-based model in the next section.

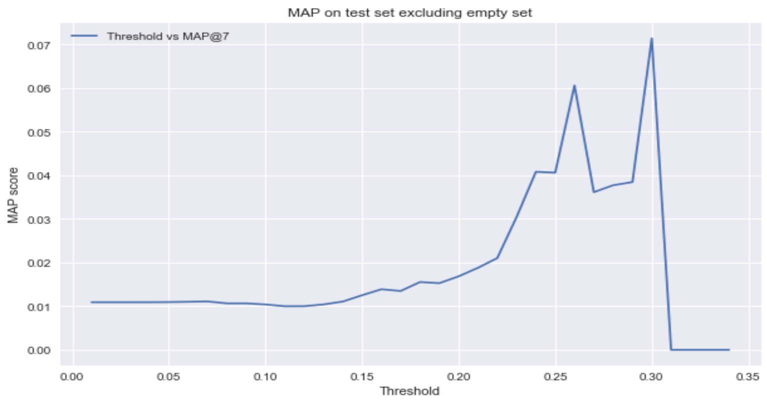


Figure 12: MAP after customer filter

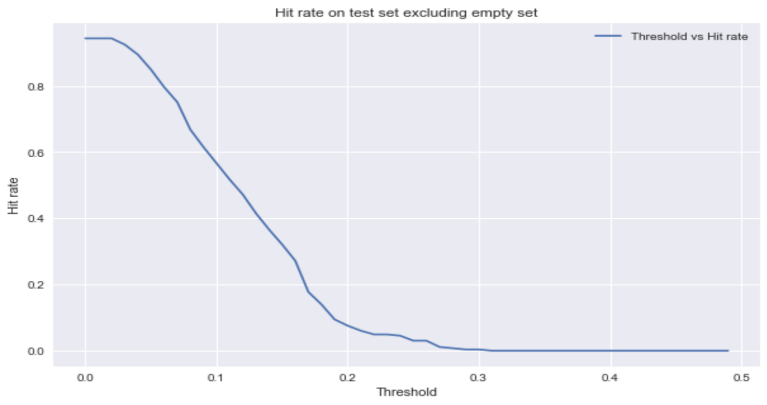


Figure 13: Hit Rate after customer filter

4.2.4 Feature Importance of XGboost Model

The feature importance is listed in Figure 14, from this graph we can find that income, age and customer seniority are the most three significant features.

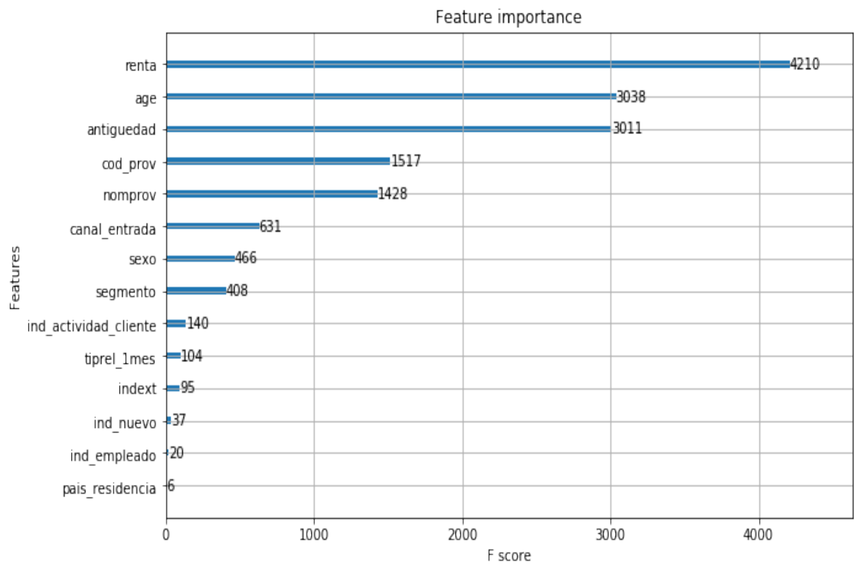


Figure 14: Feature Importance of XGboost model

5 Model Comparisons and Result evaluation

5.1 Comparison between models:

Comparing both traditional recommender and multi classification model, the result is not necessarily improved with the XGboost.

One of the advantages of XGboost is that it takes demographic information and output values with 0 to 1 probabilities. It is much easier to interpret and to make recommendations based on this result. XGboost also takes less steps from training to generating recommendations given data of same size. However, compared with similarity model, the purchase behavior may not be highly related with demographic information. Given that possible reason, directly comparing purchase behavior in similarity score may perform better than XGBoost. The tables shows the advantages of each model:

Model	Advantages
XGBoost	1. Take demographic information 2. Easy interpret with probability score 3. Less procedures (few steps from training to recommendations) 4. Relatively short running time
Similarity model	1. Directly calculate similarity of purchase behaviors 2. Easy interpret algorithm than ML models

5.2 Result Evaluation

Off-line evaluation of products recommender system is not easy. First of all, the recommendation is not necessarily a single output but could be a long ranked list. Mean Average Precision is used instead of the simple Precision and Recall. Another issue with measuring performance of recommendation system is that it is not only about accuracy, in real business world, other metrics such as diversity, coverage ratio, serendipity and novelty are also important given the purpose of recommendation. Therefore, the model choice based on MAP and Hit Rate could be vary.

From our data result, less recommendation items for each user means a relative higher MAP score but with a lower Hit Rate. On the contrary, providing all 7 recommendations results in a extremely high Hit Rate but with a poor MAP score. Also another pitfall of high accuracy score is that less valid recommendations are made if we set up a high threshold. For example, in our XGBoost model, there are only 265 valid recommendations in our validation dataset among 10000 users. If the threshold is set up to 0.4, only 10 to 20 recommendations are actually made based on our prediction, which is inefficient andoose useless in real world recommendation system.

Therefore, other metrics should be introduced to leverage the model decisions such as the efficiency rate that defined as the percentage of valid recommendation(not null) among total number of recommendations. Therefore, higher MAP with less recommendation items will have a lower efficiency rate.

In addition, based on attributes(tags) of items, the diversity of recommendations could also be controlled. For example, diversity measures the variety of the recommended items. There could be a content vector of the items like [0.2, 0.05, 0.03, 0.5, 0.1] which means for 4th feature such as long-term and low risk product accounts for 50% features of the recommendations. The 1st feature such as short-term share product is expressed by 20% of the product recommendations. Therefore, an overall score of the diversity could be calculated based on this vector and serves as another criteria choosing models.

6 Business values and future work

6.1 Business Values

The recommendation system has been widely used among all kinds of financial and electronic commerce companies, which provides incredible enhance on product selling. With the delicate insights and analysis of the customers' demographic and purchase records, the recommendation systems could increase the probability of discovering interested new products, enhance the frequency of users product purchase and save lots of time for customers finding target products.

In terms of the business use of recommendation system, on-line evaluation such as, A/B test, ROI (Return On Investment), CTR (Click through rate) , CR (Conversion Rate) etc. are usually used to measure the performance of a recommendation algorithm.

In order to leverage business decisions for different purposes, we are going to calculate more different types of metrics on testing the performance of our model in the future work.

6.2 Future Work

For future work, we are going to evaluate recommend efficiency rate, diversity score and profitability score for our recommendations. For example, for each financial items with a name, we can make feature tags such a risk, period, and profitability based on financial knowledge from other data resources. As a result, our ideal metric is a weighted score of MAP, hit rate, diversity, efficiency rate and profitability.

Therefore, when making recommendations based on different models, we could find the parameters that maximize the score. In addition, if we can have access to more business data such as Conversion Rate, Return On Investment we can have a more granular score of measurement.

So far in this project, we mainly focus single recommender model. To make it more like a "system", mixture of models is also possible. One of the ideal procedures could be choosing recommend items from similarity recommender, then re-calculate emission score based on probability prediction from the classification model, which is a weighted score combining different models or the relationship between scores could be even a non-linear function.

7 Acknowledgement

This project was supported and advised by Bonnie Ray ,who is the Vice President of Pegged Software and previous director of Cognitive Algorithms at IBM Research. We would like to express our thanks to her who provided insight and advice that greatly assisted this project.

References

[1] H. Jegou, F. Perronnin, M. Douze, J. Sanchez, P. Perez, and C. Schmid. (2012) *Aggregating local image descriptors into compact codes.*

[2] F.Perronnin & C.Dance. (2007) *Fisher kernels on visual vocabularies for image categorization. In CVPR, 2007.*

[3] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. (2014) *Imagenet large scale visual recognition challenge* **15**(7) :arXiv:1409.0575.

[4] Zhou, Z., Zhao, G., Hong, X., & Pietikainen, M. (2014). A review of recent advances in visual speech decoding. Image and vision computing, 32(9), 590-605.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun . (2015). *Deep Residual Learning for Image Recognition.* arXiv:1512.03385

[6] Lei Zhang, Yangyang Feng, Jiqing Han, Xiantong Zhen. (2015). *REALISTIC HUMAN ACTION RECOGNITION: WHEN DEEP LEARNING MEETS VLAD*

[7] https://www.kaggle.com/c/santander-product-recommendation/discussion/25579