Petr Shypila

Petr_Shypila@epam.com

12/18/2016

Santander Competition Report

This paper briefly describes details about Santander Product Recommendation and techniques which were used to solve product prediction problem.

1. Competition Details

October 26th 2016 Santander Bank launched a competition hosted by Kaggle the goal of which was to predict which products bank's customers may buy in next month based on their past behavior. The data starts at 2015-01-28 and has monthly records of products a customer has, such as "credit card", "savings account", etc. The goal was to predict 7 most probable additional products a customer will get in the last month, 2016-06-28, in addition to what they already have at 2016-05-28.

The evaluation metrics used is Mean Average Precision 7 (MAP@7)

$$MAP@7 = \frac{1}{|U|}\sum_{u=1}^{|U|} \frac{1}{min(m,7)} \sum_{k=1}^{min(n,7)} P(k)$$

where |U| is the number of rows (users in two time points), P(k) is the precision at cutoff k, n is the number of predicted products, and m is the number of added products for the given user at that time point. If m = 0, the precision is defined to be 0.

The data structure is following:

Column Name	Description
fecha_dato	The table is partitioned for this column
ncodpers	Customer code
ind_empleado	Employee index: A active, B ex employed, F filial, N not employee, P pasive

pais_residencia	Customer's Country residence
sexo	Customer's sex
age	Age
fecha_alta	The date in which the customer became as the first holder of a contract in the bank
ind_nuevo	New customer Index. 1 if the customer registered in the last 6 months.
antiguedad	Customer seniority (in months)
indrel	1 (First/Primary), 99 (Primary customer during the month but not at the end of the month)
ult_fec_cli_1t	Last date as primary customer (if he isn't at the end of the month)
indrel_1mes	Customer type at the beginning of the month ,1 (First/Primary customer), 2 (co-owner),P (Potential),3 (former primary), 4(former co-owner)
tiprel_1mes	Customer relation type at the beginning of the month, A (active), I (inactive), P (former customer),R (Potential)
indresi	Residence index (S (Yes) or N (No) if the residence country is the same than the bank country)
indext	Foreigner index (S (Yes) or N (No) if the customer's birth country is different than the bank country)
conyuemp	Spouse index. 1 if the customer is spouse of an employee
canal_entrada	channel used by the customer to join
indfall	Deceased index. N/S
tipodom	Addres type. 1, primary address
cod_prov	Province code (customer's address)
nomprov	Province name
ind_actividad_cliente	Activity index (1, active customer; 0, inactive customer)
renta	Gross income of the household
segmento	segmentation: 01 - VIP, 02 - Individuals 03 - college graduated
ind_ahor_fin_ult1	Saving Account
ind_aval_fin_ult1	Guarantees
ind_cco_fin_ult1	Current Accounts
ind_cder_fin_ult1	Derivada Account
ind_cno_fin_ult1	Payroll Account
ind_ctju_fin_ult1	Junior Account
ind_ctma_fin_ult1	Más particular Account
ind_ctop_fin_ult1	particular Account
ind_ctpp_fin_ult1	particular Plus Account
ind_deco_fin_ult1	Short-term deposits
ind_deme_fin_ult1	Medium-term deposits

ind_dela_fin_ult1	Long-term deposits
ind_ecue_fin_ult1	e-account
ind_fond_fin_ult1	Funds
ind_hip_fin_ult1	Mortgage
ind_plan_fin_ult1	Pensions
ind_pres_fin_ult1	Loans
ind_reca_fin_ult1	Taxes
ind_tjcr_fin_ult1	Credit Card
ind_valo_fin_ult1	Securities
ind_viv_fin_ult1	Home Account
ind_nomina_ult1	Payroll
ind_nom_pens_ult1	Pensions
ind_recibo_ult1	Direct Debit

2. Problem solving

2.1. Feature preprocessing

This section describes the flow of solving the problem. Although during development many techniques were used, this report mostly describes the final approach. The code could be found on GitHub under /notebooks folder.

One of the biggest problem for this problem is that there is no test data for a month for which prediction should be made. Fortunately there is such data for a previous year. So in this approach I am going to train classifier on a data for a last year and same months. First for all we start with basic data exploration. We look on a features, their values, distribution and how many data is missing. Then we preprocess features to prepare them for classifier. At the final step we test several classifiers, tune it by applying cross-validation and choose the best model.

We use iPython Notebook editor since at allows easy way of changing code on the fly as well as partial code testing. The raw data presented in CSV files. To allow easy grouping, sorting and filtering techniques Pandas data storage will be used.

Now let’s briefly talk about feature preprocessing.

Features like ‘sexo’, ‘indersi’ and ‘indext’ contains only binary values. However, they are presented by characters, so here we just replace them with integers 0 and 1.

Features like ‘ind_empleado’, ‘indrel_1mes’, ‘tiprel_1mes’, ‘segmento’ are similar with previous one. They also represent some categorical values. However, what is important, is that each of these features have more than 2 unique values. In such cases one-hot encoding could be used. But after surfing on Kaggle forums and personal experimenting I decided not to do it. I preferred to replace each category with some value where similar categories have closer numbers. This approach didn’t reduce prediction accuracy or even have improved it. One more advantage for this decision is that we kept our model simpler.

Now let’s talk about ‘nomprov’ and ‘cod_prov’. Basically, one of those features could be removed because it is just a mapping between province code and its name. Here I remove ‘nomprov’ just after the loading. What else can we do with ‘cod_prov’? Although this feature contains numbers, they are also categorical ones. And all of them have same weight and ideally one-hot encoding should be used. However, since amount of unique values is high it will make our model much more complex. So, I decided to replace original indices with ascending order of province mean income. The higher mean income province has the higher index value will be.

Feature ‘renta’ represents customer’s income. It has continuous type. Here I decided to replace actual customer’s income with a group which customer belongs to. For each province I calculated .2, .4, .6, .8 ‘renta’ percentiles and replaced them with each customer’s actual ‘renta’ value. If customer’s income was less then .2 percentile in his province he got a 0 group. 1 if higher than .2 but lower than .4 and so on. If customer didn’t provide ‘renta’, NaN value used.

Same approach was used to split customers by age. I decided not to split them by decade (0-10, 11-20, etc.) but by their expected life period (0-25, 26-30, 31-40, 41-60, 61+). Here I expect that customers below 25 are students, 26-30 are graduated young people who just start their independent life. Probably they also got married. Their needs supposed to be different comparing to people younger than 25. If customer’s age is not presented NaN value will be used.

The set of products represented by binary values(1/0). 1 if customer used product in this month, 0 otherwise. So, it doesn’t require any preprocessing.

2.2 Data classification

During work on a problem many techniques were used. However, I finished with XGBoost implementation of Gradient Boosting. XGB implementation scalable, robust to missing data and it has ability to perform cross validation. We trained our model on a data for 2015-05 and performed testing on 2015-06. Also, we used only those users for training who has added at least one product in 2015-06. Users who didn’t add any new product in 2015-06 doesn’t help us anyhow, since if we won’t offer user any new product it won’t improve our score. Even if he won’t get any new product.

To find optimal XGB configuration we perform cross-validation provided by sklearn package. This part of code is missing in final solution. I left only optimal parameters. For training XGB classifier I partially followed instructions from this webpage, especially which describes how to optimize classifier. Since XGB works only with numeric data we store all labels in array and replace labels with its index in that array. Here I use *multi:softprob* loss function to get probability for each label, so we can get 7 most popular and concatenate them into a single string which is a set of products which customer may buy in next month.

My solution predicted products with MAP@7 score 0.0265501. Current best score is 0.0309991.