Santander Product Recommendation

Team Name: TAMEthatBENCH

Agam Kashyap
IMT2018004
International Institute of Information Technology, Bangalore
agam.kashyap@iiitb.org

Soham Kolhe
IMT2018073
International Institute of Information Technology, Bangalore
soham.kolhe@iiitb.org

Tanishq Jaswani
IMT2018077
International Institute of Information Technology, Bangalore
Tanishq.Jaswani@iiitb.org

***Abstract*—This is a detailed report about our work on building an effective model for predicting the products that a given customer of the © *Banco Santander* will buy or discontinue a product offered by the bank.**

***Index Terms*—Autocorrelation, Lags, Confidence Interval, Weighted Average, LGBM, SGD Classifier, Logistic Regression**

I. Introduction

As the economy is shifting to a more digital version of its old-self, the competition in the market is increasing day-by-day. Providing a better user experience is a key factor for taking the lead. Thus, every commercial organisation has inculcated the feature of providing digitalised product recommendations to its users.

According to a research[1], product recommendations account for up to 31% of E-Commerce revenues . These recommendations are a result of complex algorithms which take into consideration the behaviour of the customers such as- the past purchases, the products viewed, search engine history and so on. *Banco Santander* also provides a wide variety of products to its customers.

The final model being generated in this project is aimed to act as a means of building a better product recommendation system for the Santander Bank, which shall help both the customers, in better selection of a plan/scheme/product of the bank, and help enhance the revenue of the bank.

Note: All scores mentioned in the report are on the public dataset, unless mentioned otherwise.

II. Problem Statement

The issue being faced by the Santander Bank was that the recommendation system gave a large number of recommendations to a small number of customers, while many others rarely received any. This resulted in a uneven customer experience. The challenge given to us leads to the solution of this problem. We are supposed to predict the top 5 products that a customer will either buy or drop in the upcoming month, given several features describing the characteristics of a customer and their past behaviour.

[1]Conducted by Barilliance Research on 300 randomly selected customers

An enhanced recommendation system, will be essential to enhance the user experience for the customers and in turn beneficial to the bank itself.

III. Dataset

The dataset provided to us is a part of the dataset used in the original challenge **Santander Product Recommendation** hosted by Banco Santander hosted on Kaggle.

It contains of 16 months of data beginning from the 28^{th} of **January, 2015** and going up to the 28^{th} of **April, 2016**. Excluding the *fecha_dato* and *ncodpers* we have 22 features for each of the customers. We have been given 24 product columns consisting of binary values for which we have to predict whether they will be dropped or picked up in the upcoming month i.e. **May 2016**. Overall we have 12715856 records for 951952 unique customers.

Since it has target columns with binary valued features it is a simple binary classification problem.

IV. Preprocessing and Feature Selection

We did some extensive preprocessing of the data initially, but got better outputs from simpler processing. But the following observations were very crucial in understanding the importance of certain features and deciding on an appropriate mode of data cleaning.

A. Features

Initially we looked at the customer ids' distribution (Fig. 1) and realised that there are a few customers which come later in time into our dataset. This resulted in the assumption that for such customers, the value corresponding to the products in the months they weren't present can be taken to be 0.

Looking at the distribution of the age of customers, we can see that it is a bimodal distribution firstly (Fig. 2). Next we see that there are customers whose ages are less than 18 as well. Seeing that there were customers aged as young as 2 or 3 years old was a curious factor, but looking at the products offered by the Santander Bank[2] this seemed good enough. So based on this, we decided to keep the lower bound on age as

[2]Santander offers the facility of 1—2—3 Mini Current accounts for children under the age of 13. https://www.santander.co.uk/personal/current-accounts/123-mini-current-account

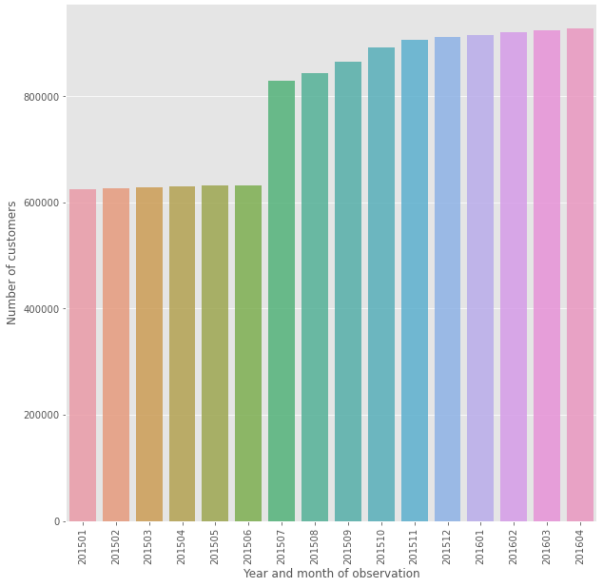

Fig. 1. Distribution of Customers across months

14, while the upper bound as 90. We then normalized this data using the MinMaxScaler provided by sklearn.

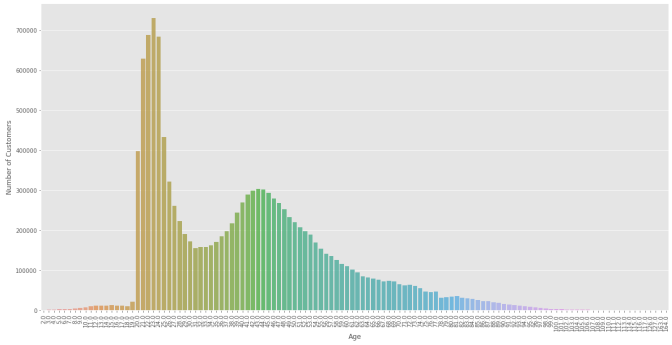

Fig. 2. Age distribution - a B-modal plot

There is another trend that we notice when it comes to *fecha_alta* i.e. when the customer became first holder of a contract with the bank. Look at Fig. 3. The number of customers seems to peak every year near the months of July to December. This seemed like an important time frame after seeing this trend, but as we report in VI, this time frame turned out to have little to almost no effect on increasing the accuracy.

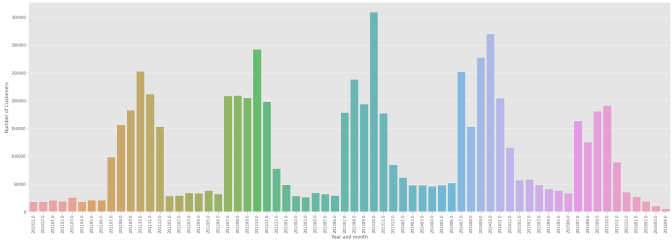

Fig. 3. Age distribution - a B-modal plot

For the income feature, *renta* we took motivation from most of the discussions on kaggle, where we filled the null values of income with the medians for each province. This was then standardized using StandardScaler from sklearn library. For all of the remaining numerical features, post reduction to most basic possible data type, we did median imputation.

Coming to the categorical features, the features where we did something different than mode imputation were *pais_residencia* and *nomprov*. *Pais_residencia* had 119 unique values of countries, while *nomprov* has 53 unique values. For both of these columns we decided to select just the top 10 of the categories in them and dump all the other into a single category, because the count of the remaining categories in both cases was relatively very low compared to the top categories. For the null values in case of the categorical features, we did mode imputation. In the feature *tiprel_1mes*, we changed the datapoints with the category **R** into **I** because the count of **R** is negligible. All the categorical features were then one-hot encoded.

B. Feature Selection

For selecting the best features, we used a pyramid based approach, wherein, we start with a single feature and go on selecting more features and checking how the accuracy turns out to be. The following points describe the process in short. All of these are from a model based on SGD classifier:

- Renta, Nomprov : **0.02678**
- Renta, Nomprov, ind_nuevo : **0.02671**
- Renta, Nomprov, ind_nuevo, segmento, ind_actividad_cliente, pais_residencia, ind_empleado, sexo, tiprel_1mes, indrel_1mes, antiguedad : **0.02773**
- Renta, Nomprov, ind_nuevo, segmento, ind_actividad_cliente, pais_residencia, ind_empleado, sexo, tiprel_1mes, indrel_1mes, antiguedad, indrel, indext : **0.02779**
- Renta, Nomprov, ind_nuevo, segmento, ind_actividad_cliente, pais_residencia, ind_empleado, sexo, tiprel_1mes, indrel_1mes, antiguedad, indrel, indext, indresi, indfall, canal_entrada : **0.02803**

For some of the other features the score decreased, but for simplicity we have displayed only the increment. So utilising this method, we were able to come up with a set of features which provide the best score.

There were 27734 rows which had almost all the columns as empty. So we decided to drop these columns. Further we dropped the features *tipodom*, *ult_fec_cli_1t*, *codprov* and *fecha_alta*. *Tipodom* had all the values as the same value 1, if it wasn't null, while dropping *ult_fec_cli_1t* was an intuitive choice. *Codprov* was redundant as it had the same information as the *nomprov* feature.

C. Target Variables

The trends were different for different kind of products.

1) Some products such as e-accounts and Current Accounts (Fig. 4) have sigmoid-like increase in the product count as we approach May of 2016. Initial inference from this

was to not include this in the count of customers taking or dropping products, but the methods tried on separate products separately didn't give good enough results as we'll see later.

2) Products like Particular Plus accounts(Fig. 5), Medium-term deposits, Mortgage have a linearly decreasing demand with time.
3) This trend is only seen for the Funds where after increasing until the month of May,2015 it becomes almost constant, only slightly fluctuating.
4) There are some other products such as Credit Cards (Fig. 7), Junior Account, Más particular Account which achieve maximas or minimas in between and then return to approximately same count as they had in the initial time period.
5) Few products don't show any trends, such as Pensions and Payrolls, but overall we do see that their count has increased, though many highs and lows in between.

While calculating lags, there were cases when for a given customer, there was no previous month's record, for such products we imputed the value as 0, because it a safe case to assume that the person didn't have that given product in the past month. Similarly, the test dataset also had some new customers, so for them as well, we assumed that the recent products they had were none.

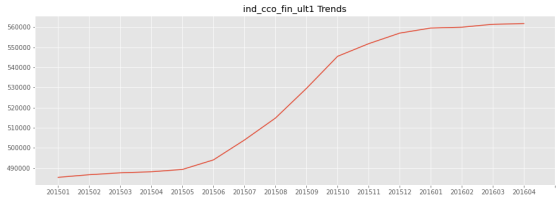

Fig. 4. Current Accounts

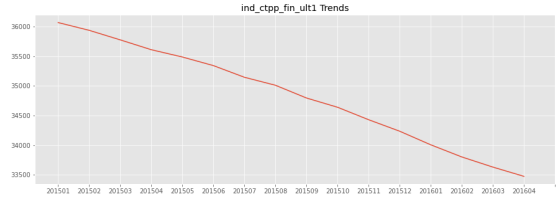

Fig. 5. Particular Plus Accounts

V. Lags and Autocorrelation

Given that our data is a time series, autocorrelation is one of the most important things to analyse.

Defn: Given measurements, $Y_1, Y_2, ..., Y_N$ *at time* $X_1, X_2, ..., X_N$*, the lag* k *autocorrelation function is defined as*

$$r_k = \frac{\Sigma_{i=1}^{N-k}(Y_i - \bar{Y})(Y_{i+k} - \bar{Y})}{\Sigma_{i=1}^{N}(Y_i - \bar{Y})} \quad (1)$$

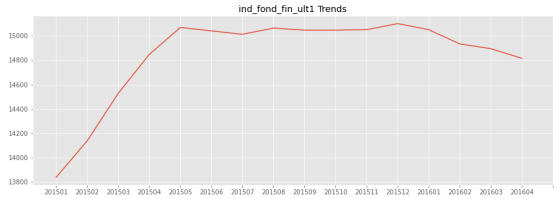

Fig. 6. Funds

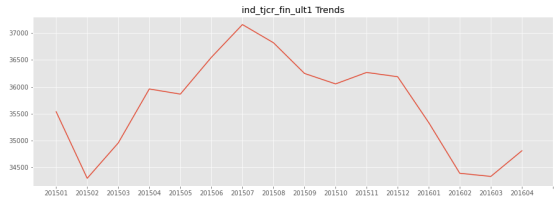

Fig. 7. Credit Card

It tells us if a given time series is linearly related to a lagged version of itself. So this means, it will help us find out if there are any patterns in the time series. Those time frames or 'lags' with high autocorrelation coefficient would imply that the series tends to repeat itself in that time frame. An example of the autocorrelation function plot is shown in Fig. 3, where the x-axis contains the lag value while y-axis contains the autocorrelation coefficient. The conical area defines the confidence interval.

Confidence Interval gives us an estimate of the range in which the unknown parameter would fall in. Since the confidence interval depends on the variance of the different samples, in the dataset, we can see in the fig[] the region which looks like the tip of the cone, i.e. has the least interval size, has the least variance with the original lag.

Why does the April 2016 data alone provide a better result than that combined with that of 2015 around the same time, both of which compared to the later half of the year 2015 perform exceptionally better?(Table I) This is due to the before explained concept of autocorrelation. Figure 1 has been drawn with 90% confidence interval. As we can see, the only value of the autocorrelation coefficient that lies beyond the conical shaded area is for lag 1 and lag 2 only. For the lags - 10, 11, 12 which fall in the time period of April 15, May 15, June 15 is negative and inside the confidence region. But also notice, that this sine curve gets its minima around the range of lag 10 to 11. So, the negative autocorrelation is still high enough to provide significant temporal information to our model(As we will see in the section VI)

But, look at the lags from 5 to 9(which showcase relation with the second half of the year 2015). They are very low in the value, almost negligible around 5 and 6, and hence, they being so less of importance, including that as a part of our time frame did not amount to anything positive in terms of improvement.

TABLE I
TIME PERIOD COMPARISON

Time-Frame	Accuracy
201503 - 201507	0.02782[1]
201503 - 201510	0.02825[1]
201503-201512	0.02837[1]
201503-201512 + 201604	0.02832[1]
201604	0.02881[1]

[1]Calculated using SGD based model, alpha=10^{-2}

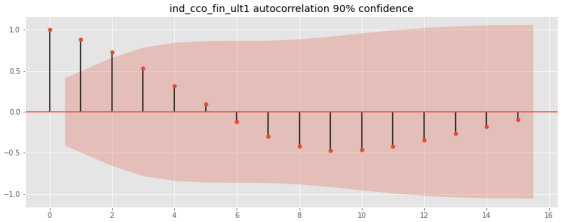

Fig. 8. Lags vs Autocorrelation coeff

VI. TRAINING AND RESULTS

One of the most crucial decision that we had to make was that of a time frame that gave the best result. Our initial instinct was to select months which were around the month of May, because customers who purchased a certain product around that time in 2015, might purchase, or even drop products around the same time. This proved to be a good starting point. As discussed in Section V, we tried out different time frames. We'll discuss the time frames further for each model.

A. SGD Classifier

After having tried out logistic regression and seeing the amazing increment it was giving, we decided to try SGD classifier given that it provides us with much more flexibility with the hyper parameters. The best model that we got from here had the following parameters:

- *alpha* : 10^{-2}
- *loss* : log
- *max_iter* : 10000
- *shuffle* : True

After realising that the time frame which gave us the best result was just the month of April,2016, we kept on working with it. After getting a cap of accuracy on SGD, we started experimenting with lags(Table II).

TABLE II
LAGS ACCURACY COMPARISION

Lags	Accuracy
1	0.03105
1,2	0.03442
1,2,3	0.03473

An interesting idea at this point was to modify the model and make it unique to a given product. If you look at the auto-correlation plot of all the different products, the autocorrelation coefficients which are significant, are different for each. So, we made separate models for each of the products with the training data containing number of lags, as determined by the plots. This didn't prove to be fruitful as the accuracy with this fell down to **0.03160**. But this did give rise to another idea, which then helped us push through.

We noticed that the accuracy for the time frame of 201503-201506 was almost comparable to that of 201604 alone. So we decided to use a weighted average of those two models. *Giving 0.3 weight to 201503-201506 time frame, with 2 lags taken and remaining 0.7 to 201604 with 3 lags taken we got an increase in accuracy to* **0.03625**.

B. Light GBM

LGBM training picked up from where we left off in SGD Classifier.

Time Frame 1	Lags 1	Weight 1	Time Frame 2	Lags 2	Accuracy
201503-201506	1,2	0.3	201604	1,2,3	0.04379
201503-201506	1,2,3	0.3	201604	1,2,3	0.04415
201503-201506	1,2,3	0.3	201602-201604	1,2,3	0.04428

**Assume the weight of time frame 2 to be* (1- Weight 1)

Another thing that we tried was using the data from complete dataset but in a similar weighted manner, i.e. Lag1,2 for [201503-06] with 0.25 weightage, Lag 1,2 for [201602-04] with 0.65 weightage and Lag1,2 for [201507-201601] with 0.1 weightage. Though this couldn't give any better results than before.

Excluding the hyperparameter tuning, the next big push came by doing feature selection. This involves utilising the *feature_importances_* feature of LGBM to figure out the least important features in our training data set, remove them, and retrain the model over this new training dataset.

The final push that we got, post tuning the LGBM model, was by introducing another month into the time-frame of the 2015 year. i.e. **Timeframe 1: 201503-201507, weight: 0.3, lags: 1,2,3,4**
Timeframe 2: 201602-201604, weight: 0.7, lags: 1,2,3,4,5 which gave an accuracy of **0.04603**.

The parameters of the best model were as follows:

- *boosting type*: gbdt
- *max depth* : -1
- *objective*: binary
- *learning rate*: 0.1
- *num iterations*: 200
- *max bin*: 512
- *subsample*: 1
- *subsample freq*: 1
- *min split gain*: 0.5
- *min child weight*: 1
- *min child samples*: 5
- *scale pos weight*: 1
- *verbosity* : 1

The scoring metric for this competition was MAP@5 that required us to predict a set of 5 products for each customer in order. For achieveing this, we used *predict_proba* to find

the probabilities of a given customer to belong to that class i.e. of the corresponding value of the product feature being 1. But given that our problem was to return the products that a person would either take or drop, we had to check the previous month's values as well. The following code handles this:

```
if product value in prev month == 1:
    prediction_customer = 1 - (predict_proba from
    model)
else:
    predict_proba from model
```

The modified probabilities were then sorted in ascending order and the corresponding top 5 products were returned.

C. Shortcomings and Scope of Improvement

The following points display some more ideas/faults that we couldn't notice or implement during the competition, which could've given a better output.

1) The accuracy that we got with the whole dataset, as mentioned in VI, was *0.04296* which was much greater than the accuracy obtained by excluding that time frame(*more than **18%** increment*). Our fault was that we compared this model with the model that didn't have the same number of lags, which led to the false conclusion of it being a worse model.
2) The best model that we got, had only been the first, i.e. we weren't able to experiment further. We believe testing out a broader time period for the year 2015, and tuning the LGBM model according to the 'dart' booster, can result in a more accurate model.
3) Testing out other ensemble techniques such as stacking could make a better model.

VII. Conclusion

Our model does have a temporal logic to it, i.e. it is able to recognise the pattern of customers, with a MAP@5 score of 0.04567 on private dataset. As mentioned above, the model does have a huge scope of improvement and given some more time, we could've built a better model. But, through this contest, we were able to understand the working of a time series and the intrinsic concepts underlying it. Time series problems, such as the stock market prediction had always been of deep interest and this project helped in better understanding of it. This project was further a motivation to understand the ARIMA model and wish to work on it in the future.

Acknowledgment

We would like to thank our amazing TA Tejas Kotha, for organising this project and competition, and helping us out with any of our doubts or problems that we were facing. The brain-thrust activities were an amazing method and they helped us learn a lot of stuff and made this whole competition into a vast learning experience. We would also like to thank the TA Shreyas Gupta for his guidance during the project and for sharing his experience with time series. We would like to give a huge thanks, along with congratulations to the team Breaking Code- Aarushi, Ayush Sawlani and Ayush Mishra, for helping us out in finding the tiny issue which was proving fatal to us. We would also like to thank all the TAs of the course as well for the tutorial sessions which they organised which were influential in helping us select the perfect model for a given situtation. We would love to thank our Professors, Prof. Raghavan and Prof. Neelam for their lectures, which made the subject very clear and piqued our interest in the field.

References

[1] Handbook of Statistical Methods (NIST SEMATECH) https://www.itl.nist.gov/div898/handbook/eda/section3/eda35c.htm
[2] Kaggle kernel by Hachem SFAR https://www.kaggle.com/hachemsfar/naive-bayes
[3] Posts by Evgeny Patekha, Tom Van de Wiele on original contest https://www.kaggle.com/c/santander-product-recommendation/discussion