Santander Product Recommendation
TEAM AAP

Aarthi Sree mundla
IMT2018046
Aarthi.Sree@iiitb.org

Abhigna Banda
IMT2018002
Abhigna.Banda@iiitb.org

Pavan Sudeesh.Peruru
IMT2018517
Pavan.Sudeesh@iiitb.org

1. Introduction

The Internet gives us so much information that sometimes confuses us. Recommendation system tries to overcome this problem.

A product recommendation system is a software tool designed to generate and provide suggestions for items or content a specific user would like to purchase or engage with. It filters the data using different algorithms and recommends the most relevant items to users. There are many different types and uses for recommender systems.

The recommender systems are broadly categorized into two types: Content based and collaborative filtering.

An increasing number of online companies are utilizing recommendation systems to increase user interaction and enrich shopping potential. Big companies like Amazon, Netflix, Linkedin also use recommendation systems. It gives advice to the user and helps him in choosing. Even if a new customer joins, based on popularity of items and personal information of the customer, he is recommended to products. It makes choosing things in life easy.

This project is SANTANDER BANK RECOMMENDATION. Aim of the project is to predict top 5 products which are most likely to be added or removed from their original product list. It is also a product recommendation system. We predict the product that a customer may desire in the upcoming month and suggest or offer discounts on such products that boosts up our sale.

2. DATASET GIVEN TO US

We were provided with 1.5 years of customer behavior data from Santander bank. Data was recorded on the year of 2015(1-12 months), 2016(1-4 months). The data set given to us has 48 columns in which 24 are the product columns. For every customer there will be a unique Id named "ncodpers" . Every month a new customer can either be added or deleted. Each customer can buy or drop any product. The product columns have only 2 values(0 or 1). 1 means the product has been taken for the respective month and 0 being the product was not taken for the respective month.

The last 24 columns are products. The train data had 12715856 rows and 48 columns while the test had 931453 rows and 24 columns. We have named our train data frame as bank_data and test data frame as df_test. As the number of rows were very large, it was hard to deal with data. So we divided our data based on fecha_dato. Fecha_dato had 16 months. We have taken only 1,2,3,45,6,7,8 months. We had taken many combinations of months but this combination(5,6,7,8) yielded the best result.

3. DATA PREPROCESSING

3.1. Filling Of Null Values

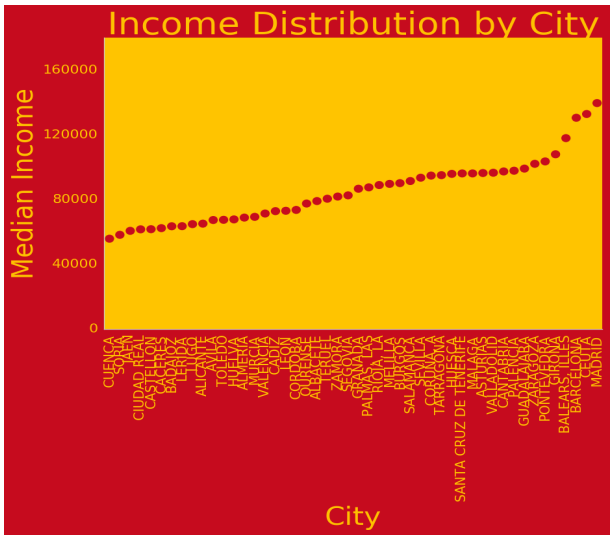
The cause of missing values can be data corruption or failure to record data. The handling of missing data is very important during the preprocessing of the dataset as many machine learning algorithms do not support missing values. A model trained with the removal of all missing values creates a robust model.

The bank_data had null values in 22 columns. The df_test had null values in 10 columns.
In bank_data: Columns like ind_empleado, pais_residencia, ult_fec_cli_1t , ind_nuevo, indresi, tipodom, nomprov, tiprel_1mes, indext, conyuemp, canal_entrada, indfall, sexo, segmento had null values. As they are of object data-type, we have replaced them with a new category termed as "UNKNOWN" .
Columns like indrel, ind_actividad_clienta are float data types and hence they were filled with their medians. In columns like indrel_1mes the null values were replaced by their modes.
COD_PROV
There were some product columns that had null values. The null values in them were replaced by '0' as we didn't found any correlation with any other columns so the best way to replace to replace the null values to create a new unknown province and name that cod_prov as Zero.

In df_test, null values were there in sexo, ult_fec_cli_1t, indrel_1mes, tiprel_1mes, conyuemp, canal_entrada, cod_prov, nomprov, renta, segmento. The null values in df_test were replaced in a similar way as they were done in bank_data.

In df_test, null values were there in sexo, ult_fec_cli_1t, indrel_1mes, tiprel_1mes, conyuemp, canal_entrada, cod_prov, nomprov, renta, segmento. The null values in df_test were replaced in a similar way as they were done in bank_data.

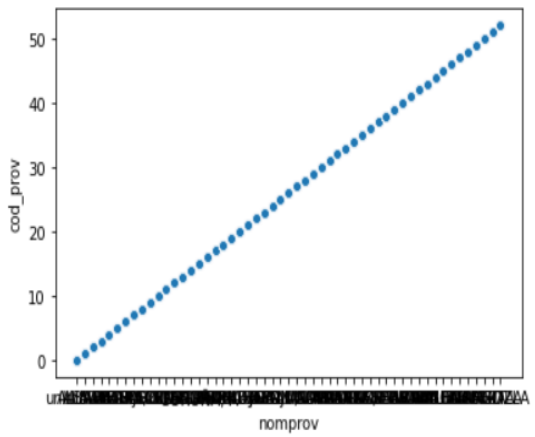
In a specific column named renta we found out that every province has a different means of renta and there is a significant difference so we filled the null values of renta with means of their specific provinces.



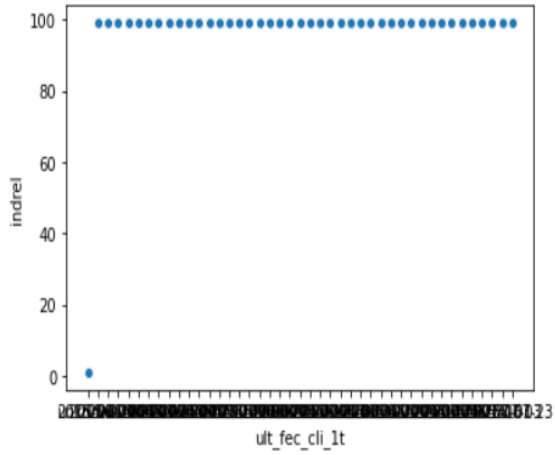
There are some columns like renta, age which means their income and age of the person which has numerical values. So we thought of encoding those columns also so we grouped the rentas, ages into groups like (0-50,000) as one grp of ppl, (50,000- 1,00,000) as other groups. In that way we divided entire rentas into groups. For age ; (0-18) as one group (18-28) as one grp and so on. In this way we encoded the numerical columns.

3.2. Removal of unnecessary columns

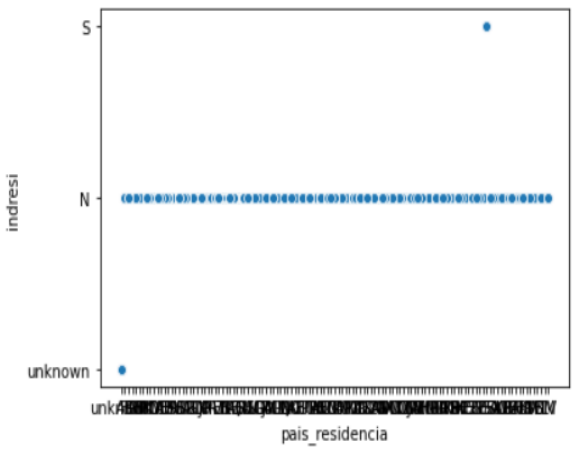
Our dataset is huge and has many unnecessary columns. We can remove those columns as they do not contribute to the training of our model.
The columns we have removed are:
nomprov : nomprov is the name of province and codprov is the unique code for the province. It is sufficient to have either of them. So we have removed nameprov.
Here we can there's 1 cod_prov for each nom_prov.So we can see there's a strong correlation between these



features.So we can remove nomprov.



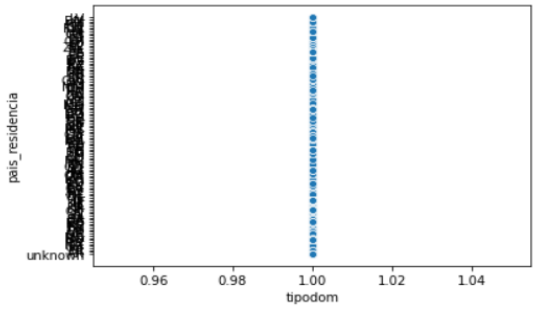
Here there's 0 if ult_fec_cli_1t is "unknown".Otherwise 100 for any thing.So these are dependent on each other.So we drop ult_fec_cli_1t.
INDRESI:



When pais_residencia is the same as bank_address. Then

it's S. For remaining it's N. So indresi is dependent on pais_residencia. So we drop indresi.
FECHA_DATO :
In the data in the start we separated the data frames based on the fecha_dato. So after dividing the data the column fecha_dato has no use so we removed that column.
TIPODOM:

tipodom :



ANTIGUEDAD:
bank_data['fecha_alta']-bank_data['fecha_dato'] in months =bank_data['antiguedad'] i.e present_date-The date in which the customer became as the first holder of a contract in the bank = customer seniority so we can remove antiguedad column
IND_NUEVO
bank_data['fecha_alta']-bank_data['fecha_dato'] if less than 6 months, then bank_data['ind_nuevo']=1 else =0 present_date-The date in which the customer became as the first holder of a contract in the bank ¡6 then new customer index is given a value 1 or else 0. So we can remove ind_nuevo column . FECHA_ALTO: We felt does not have much significance so we removed that column

3.3. Lags

The data given to is a time series data so the model predictions also will be dependent on time frame. Let us say we need to predict for data of MAY and the previous years MAY data will be correlated and the behaviour of previous month's will have an effect on the present month's data so we created some new features such as lag1,lag2,lag3 where every lag has all product columns. Lag1 shows us whether the product was taken in the previous month or not. Lag2 shows us whether the product was taken in 2 months prior to the current month or not. Lag3 represents whether the product was taken in 3 months prior to the current month or not.
IMPLEMENTATION OF LAGS:

We tried many different approaches like using loops, shift() operation .. etc but they were taking a lot of time to execute so we started exploring more and more pandas functions. After going through them we get to know that the simple operation merge can itself do the operation in very less execution time. This merge operation joins two data frames based on 'ncodpers'. We have initially divided our bank_data into groups based on fecha_dato. So each group corresponds to only one month's information. We then combined 2 successive months data frames using merge such that the one of the months products columns act as lag1 for the other month. We had done similar operations for lag2 and lag3 also.

3.4. Merging of 24 Product Columns

As the 24 products columns are sparse and occupy too much space, we merged them into 1 string.
Ex:0,1,0,1...1=0101.....1
Size before merging=190348182
Size after merging=154462869

We store the product columns to pass to our model. As our model predicts for each product we can pass each character in the string as 'int' to the model.

Lag columns are also sparse but we did not perform this operation on them because these are to be passed to the model as features. If we merge them as a string, while passing it to the model, the columns act as categorical data. So we did not perform this merge operation on them.

3.5. Encoding

The performance of a machine learning model not only depends on the model and the hyperparameters but also on how we process and feed different types of variables to the model. Since most machine learning models only accept numeric variables, preprocessing the categorical

variables becomes a necessary step. We need to convert these categorical variables to numbers such that the model is able to understand and extract valuable information. The data we are working on do not have any inherent order. Hence we used one-hot encoding. In one hot encoding, for each level of a categorical feature, we create a new variable. Each category is mapped with a binary variable containing either 0 or 1. Here, 0 represents the absence,and 1 represents the presence of that category.

We did one-hot-encoding for all categorical data type columns by using one-hot encoder(a predefined function). First we seperated all the object typed data columns and one hot encoded them using onehotencoder.

3.6. Normalization

Normalization is a scaling technique in which values are shifted and re scaled so that they end up ranging

between 0 and 1. The goal of normalization is to change the values of numeric columns in the data set to a common scale, without distorting differences in the ranges of values.

For renta and age columns we applied normalization (actual value - min_value)/ max_value - min_value.
Our score has improved slightly when we have applied normalization on the entire data.
But in the end we went with grouping them because it gave a better score for us than normalization.

4. MODELLING

The process of modeling means training a machine learning algorithm to predict the labels from the features, tuning it for the business need, and validating it on holdout data. The output from modeling is a trained model that can be used for inference, making predictions on new data points.

Our aim of the project is to predict what the customer can buy or drop in the next month. So for any product the state of it is either 0 or 1. Hence it is a classification problem. So we used classification models like:
1.)Logistic regression
2.)Naive Bayes
3.)Xgboost
4.)Light gbm

Model predicts probabilities of products of having 1.
So we assume a customer takes products if he doesn't have that in the previous month. And he can only drop a product if he has that product in recent months. For adding products
A customer can only buy products if he does not have them. Represented as a '0' meaning a customer does not have them.
So for this we need high probabilities of products having 1. However assuming they don't have that.
We sorted probabilities of not having products. We took top 5 products with high probability.
For deleting products
We have listed all the products that a customer has in his previous month. Represented as '1' - meaning customer has that product.
In having products(list of products they have) we took least probability products of retaining them as we want the state of the product to be 0.
Combining
Now we combine both adding and deleting products into 1 list. We now sort probabilities and take the top 5 products that have high probability of changing their state..

Model	Score
Plain Logistic Regression	0.00951
Plain Naive Bayes	0.01295
Naive Bayes with one hot encoding	0.01450
Light gbm with one hot encoding	0.01485
Taking data from specific Data Frames	0.01986
On applying one lag	0.02195
On taking 3 lags	0.02477

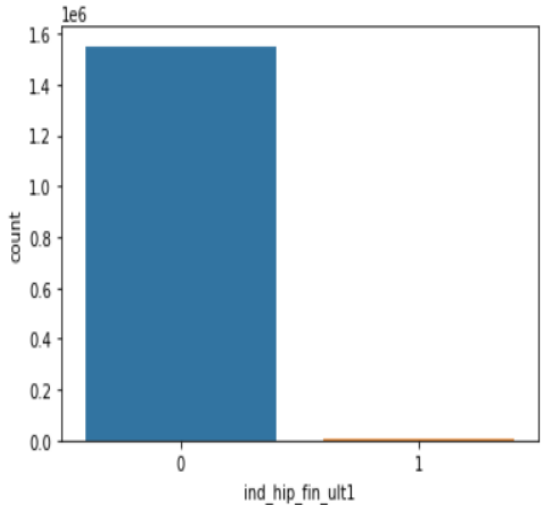
Figure 1. SCORES

5. CHALLENGES

The data set given to us was large so we had to look carefully and delete unnecessary information as much as possible.

The number of categorical columns were large and encoding all of them was very difficult. We had initially one-hot encoded and label encoded all the categorical columns except for pais_residencia and canal_entrada. We have baseN encoded these two features. But later we found an inbuilt function 'onehotencoder' that one hot encodes all the features but we had to convert it to a sparse matrix.

For lags we had initially written loops but it was taking more than 3 hours to run. This took a very long time and was very difficult to work on. Whenever we made a small change, we had to wait for hours to see its effect. So we explored and got to know about merge which reduced the running time to 15 mins.



There is so much unbalanced data. So we tried balancing.We tried 2 methods.
We tried to balance each product using SMOTE. If we balance 1 product and after another. While balancing the 2nd product, the 1st product may get imbalanced.So we felt we can't do it. We felt this may lead us to train the model with wrong data.So we tried another method.This process is not feasible.

We tried balancing by taking bank_data into 24 different tables with features and 1 product.
Ex:bank_data_product We tried balancing each table separately using SMOTE. Then we ran our model to predict. Each time we ran we immediately deleted bank_data_product. So each time there will be 1 single product table.This saves memory space. However we didn't submit this because it was taking a lot of time. So we thought of doing this in different notebooks. Each product in a different notebook. We then wanted to take probabilities of each product and combine and then sort.We couldn't do it.

Many times kaggle was crashed due to excessive usage of ram to overcome that after using of data frame we started deleting the data frames and their copies.

6. CONCLUSION

In this project we came to know about various models. We understood how important it is to pre-process the data. We have learnt ways to train our model even if data is huge and not manageable. We have learnt about the models/pre-processing of data in theory but we got to implement our learnt knowledge in this project. We have taken up this project because we were fascinated about the product recommendation system. We have learnt a lot in this regard.

7. FUTURE SCOPE

Product recommendation systems are one of the most successful and widespread applications of machine learning in business. They boost sales, revenues, click-through-rates, conversions, and other important metrics.

In this pandemic time it is difficult for banks to run but they do not want to lose out customers, for that they need to know the interest of the customers so that they can offer discounts and keep its customers while attracting the new customers. They can also advise to drop the products understanding the financial situation about the customers in these times(like maybe not enough cash, so the customers only buy necessary products). By getting to know customer needs in the next month the Bank can take required actions and make customers not to drop anything (Ex: If a lot of customers are going to delete savings accounts in the next month then the Bank can offer down payment less or 0 down payment. In this way the bank can make customers not drop products and also attract customers to take the new products. By doing this the bank can estimate customers and their requirements and make profit.

In this huge competitive world where new technologies and ideas evolve everyday, it is important to always attract our customers. By understanding the pattern in business we can expand our business.

It is an active area of research. Although recommender systems have witnessed unprecedented improvements ,a lot of research is going on to further enhance the output accuracy and improvements in all dimensions of the recommender system.

8. ACKNOWLEDGEMENTS

We would like to thank Professor G. Srinivas Raghavan and Professor Neelam Sinha and our Machine Learning Teaching Assistant Tejas Kotha for giving us guidance and support throughout the assignment. We would also like to thank other TA's who helped us in understanding the concepts. It was a great learning experience for us.

9. REFERENCES

https://www.analyticsvidhya.com/blog/2020/08/types-of-categorical-data-encoding/

https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.merge.html

https://towardsdatascience.com/feature-selection-techniques-in-machine-learning-with-python-f24e7da3f

https://realpython.com/numpy-scipy-pandas-correlation-python/

https://machinelearningmastery.com/feature-selection-with-real-and-categorical-data/

https://machinelearningmastery.com/feature-selection-with-real-and-categorical-data/

https://www.coursera.org/lecture/software-processes/model-selection-when-to-use-which-model-W387O

https://towardsdatascience.com/predictive-modeling-picking-the-best-model-69ad407e1ee7

https://blog.statsbot.co/machine-learning-algorithms-183cc73197c

https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.merge.html

https://www.shanelynn.ie/merge-join-dataframes-python-pandas-index-1/