Customer Segmentation Project

Week 9

JULY 18, 2022

Contents

1. **Group Information** ..... **Error! Bookmark not defined.**
2. **Problem understanding** ..... **Error! Bookmark not defined.**
3. **Data understanding** ..... **Error! Bookmark not defined.**
4. **Data Cleaning**..... 5
5. **Handling Missing Values**.....5-10

1. Group Information

Group Name**:** M.A.S

Specialization: Data Science

Submitted to: Data Glacier canvas platform

Internship Batch: LISUM10: 30

Group Members	**Three members**		
Name	**Email**	**Country**	**Collage/Company**
Moath Mohammed Bin Musallam	moathmusallam@gmail.com	Saudi Arabia	University of East Anglia
Andrew Kojo Mensah-Onumah	kojoakmo@gmail.com	Ghana	Data Glacier
Shaimaa Saleh Obad Al-khawlani	s.khawlany@gmail.com	Yemen	Data Glacier

2. Problem description

Most banks around the world have variant large customer base with different income levels, ages, characteristics, values and lifestyles.

XYZ bank wants to increase the production and the satisfactions of all customers categories by roll out Christmas offers to their customers.

But Bank does not want to roll out same offer to all customers instead they want to roll out personalized offer to particular set of customers. If they manually start understanding the category of customer then this will be not efficient and also, they will not be able to uncover the hidden pattern in the data (pattern which group certain kind of customer in one category).

3. Data Understanding

The existing data, which was provided by the bank, is the bank's customers data. However, the data contains many columns that will help the analytics team analyze the data and build a customer segmentation approach for the bank.

Since the data does not contain a dependent variable or (Target), We believe that machine learning (clustering) techniques would be appropriate to use for this type of data.

Size: 1000000 records, 48 columns.

- **Columns Description:**

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
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

Cleaning: is the way of get the best vision of data by dealing with missing values, duplicating, and outliers.

First, I rename the column to make then more understandable, it goes for this

Out[90]:

1	ind_plan_fin_ult1	ind_pres_fin_ult1	ind_reca_fin_ult1	ind_tjcr_fin_ult1	ind_valo_fin_ult1	ind_viv_fin_ult1	ind_nomina_ult1	ind_nom_pens_ult1	ind_recibo_ult1
0	0	0	0	0	0	0	0.0	0.0	0
0	0	0	0	0	0	0	0.0	0.0	0
0	0	0	0	0	0	0	0.0	0.0	0
0	0	0	0	0	0	0	0.0	0.0	0
0	0	0	0	0	0	0	0.0	0.0	0
0	0	0	0	0	0	0	0.0	0.0	0
0	0	0	0	0	0	0	0.0	0.0	0
0	0	0	0	0	0	0	0.0	0.0	0
0	0	0	0	0	0	0	0.0	0.0	0

To this,

Out[92]:

first_holder_date	new_cust_index	cust_seniority	...	mortgage	pensions1	loans	taxes	credit_card	securities	home_account	payroll	pensions2	direct_debit
2015-01-12	0.0	6	...	0	0	0	0	0	0	0	0.0	0.0	0
2012-08-10	0.0	35	...	0	0	0	0	0	0	0	0.0	0.0	0
2012-08-10	0.0	35	...	0	0	0	0	0	0	0	0.0	0.0	0
2012-08-10	0.0	35	...	0	0	0	0	0	0	0	0.0	0.0	0
2012-08-10	0.0	35	...	0	0	0	0	0	0	0	0.0	0.0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
2013-09-25	0.0	22	...	0	0	0	0	0	0	0	0.0	0.0	1

Then change some column values to this, to better understanding the data

```
df.cust_gender = df.cust_gender.replace({'H': 'M', 'V':'F'})
```

```
df.cust_res_index = df.cust_res_index.replace({'S': 'Y'})
```

```
df.is_foreign = df.is_foreign.replace({'S': 'Y'})
```

```
df.deceased_index = df.deceased_index.replace({'S':'Y'})
```

Checking for missing values in all columns and drop all columns have 60% and over missing values. And drop [addres_type] column because it have only one unique value.

```
#drop single unique coulmn
df.drop(['addres_type'], axis=1, inplace = True)
```

```
#dropping columns value if 60% in null

for i in df:
    null_value = df[i].isnull().sum()
    percentage = (null_value/1000000)*100
    if percentage >= 60:
        print(i," ,    this column is removed")
        df.drop([i], axis=1, inplace = True)
```

```
last_date_primary_cust  ,    this column is removed
cust_spouse_index  ,    this column is removed
```

```
# drop last_date_primary_cust, cust_spouse_index,60% or appove missing values.
# and addres_type one unique value.
```

```
df.shape
```

```
(1000000, 45)
```

Drop date column because it have just a two unique value (two date).

```
df.shape
```

```
(982162, 44)
```

```

#there are only 2 date values. It can be dropped
df = df.drop("date", axis=1)
```

```
df.shape
```

```
(982162, 43)
```

Check for duplicated raws,

```
dup = df.duplicated()
```

```
dup.value_counts()
```

```
False    646581
True     335581
dtype: int64
```

There are 335581 duplicated raws need to be dropped.

Now the data is 646581 raws after drop the duplicated raws.

```
1 dup.value_counts()
```

```
False    646581
True     335581
dtype: int64
```

```
1 df = df.drop_duplicates()
```

```
1 dup = df.duplicated()
```

```
1 dup.value_counts()
```

```
False    646581
dtype: int64
```

Screenshot of the final draft of data types

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 646581 entries, 0 to 999997
Data columns (total 43 columns):
 #   Column                Non-Null Count   Dtype
---  ------                --------------   -----
 0   cust_code             646581 non-null  int64
 1   emp_index             646581 non-null  category
 2   cust_residence        646581 non-null  category
 3   cust_gender           646581 non-null  category
 4   age                   646581 non-null  int64
 5   first_holder_date     646581 non-null  object
 6   new_cust_index        646581 non-null  float64
 7   cust_seniority        646581 non-null  int64
 8   indrel                646581 non-null  float64
 9   cust_type             646581 non-null  float64
 10  cust_rel_time         646581 non-null  category
 11  cust_res_index        646581 non-null  object
 12  is_foreign            646581 non-null  category
 13  channel_to_join       646541 non-null  object
 14  deceased_index        646581 non-null  category
 15  cod_prov              646581 non-null  float64
 16  name_prov             646581 non-null  category
 17  activity_index        646581 non-null  float64
 18  gross_income          646581 non-null  float64
 19  saving_acc            646581 non-null  int64
 20  guarantees            646581 non-null  int64
 21  current_acc           646581 non-null  int64
 22  derivada_acc          646581 non-null  int64
 23  payroll_acc           646581 non-null  int64
 24  junior_acc            646581 non-null  int64
 25  mass_particular_acc   646581 non-null  int64
 26  particular_acc        646581 non-null  int64
 27  particular_plus_acc   646581 non-null  int64
 28  short_term_deposits   646581 non-null  int64
 29  medium_term_deposits  646581 non-null  int64
 30  ind_dela_fin_ult1     646581 non-null  int64
 31  e_account             646581 non-null  int64
 32  funds                 646581 non-null  int64
 33  mortgage              646581 non-null  int64
 34  pensions1             646581 non-null  int64
 35  loans                 646581 non-null  int64
 36  taxes                 646581 non-null  int64
 37  credit_card           646581 non-null  int64
 38  securities            646581 non-null  int64
 39  home_account          646581 non-null  int64
 40  payroll               646581 non-null  float64
 41  pensions2             646581 non-null  float64
 42  direct_debit          646581 non-null  int64
dtypes: category(7), float64(8), int64(25), object(3)
memory usage: 186.8+ MB
```