https://rscheze.wixsite.com/rse-tfm2020

Context

- Sales representatives require relevant information to address a customer and build rapport.This has a deep impact on customer experience.

- This work aims to reduce the gap between a model prediction and the information required for Sales representatives to address a customer (Explanation through Explainability(1)). In this particular case for personal loans.

- Behind this work, there is not only a classification problem (whether a customer will get a personal loan or not) but also, and most importantly, why a specific person could be more interested in a specific product/service than another and the reasons behind that prediction.

- This work walks through the entire technical modelling process to be able to build a business plan with the information required for Sales representatives required on the who,how and why.

Let´s begin!!!

(1)<<Explainability/interpretability is the dregree to which a human can consistenly predict the model results>> (Kim.Been,Rajiv Khana and Oluwasanmi)

Executive Summary

- The dataframe used contains 395.937 instances and 44 features. 1.8% of the population studied have contracted a personal loan (the dependent variable).
- When the house is between 10-35 years old, with >8-year-old vehicles and clients who check bank details through their Bank App, the probability of contracting a personal loan increases.
- Comparing to the total number of clients in the population, the results obtained show that there are more clients from the Cataluña and Malaga areas.However, the success rate is the highest in the Northwest and Cataluña áreas (as shown in the previous graphic).
- 5 models were used to decide the one which better adjusted to the binary classification analysis (Logistic Regression, Decision Tree Classifier, Random Forest Classifier, XGBoost, Stack ensemble model).
- For the Business Plan, basic assumptions are created to be able to build a Business case(100k clients, average loan 6k€,margen per loan 4%, success predicted customers 22%).
- A potential outcome of 21% ROI (~350k €) per year could result if these customers are addressed: clients with <40% success rate will be contacted by WhatsApp. Between 40% and 80% by mail + phone. More 80% by mail + Branch 9.4k clients.
- 9.4k will be managed by Sales Representative. Explaining to them why this model has selected them and help them find the best manner to present it to the client is crucial. For the Local Features selection, the model used was LIME(Local Interpretable Model-agnostic Explanations).
- By using LIME, Sales Representatives can use this information(10-years-old car or if his house is 23) to offer clients the possibility to change their car without having to pay all upfront or to ask if hi/she is thinking on reforming any part of his/her house.

Although the initial information included in the dataset is real, in order to present this work and to comply with current regulations, all data has been anonymized.

Data Description

The dataset contains 400k instances and 51 features. All the information has been anonimized for the final work.

Feature	Description
cod1	Id letter identification
cod2	Numeric identification for each customer
gender	Customer gender
age	Customer age
nat_country	customer born nationality
resi_country	Customer current residency
studies	Level os studies(1-6)
work_type	Current type of work(A,O,J,P)
branch	Branch number
marital_status	Marital Status(C,V,D,O)
codseg	Customer segment code
cpper	Postal code
cprovper	Province code
clocaper	City code
autonper	Territory code
country	Country code
codine	INE code
population_cmun	City population
disporent_cmun	City average disposible money
num_cred_cards_0m	Customer number of credit cards dec-19
aveg_amount_cred_card_0m	Customer average amount on credit cards dec-19
num_current_acc_0m	Customer current accounts dec-19
aveg_amount_current_acc_0m	Customer average amount on current accounts dec-19
num_invest_funds_0m	Customer current investment funds dec-19
aveg_amount_invest_funds_0m	Customer average amount on investment funds dec-19
mortgage	Mortgage(Y,N)
num_stocks_0m	Customer current number of stocks dec-19
aveg_amount_stocks_0m	Customer average amount on investment funds dec-19
acc_0m	Number of all type of accounts dec-19
mobile_0m	Number of times log in online banking with mobile dec-19
pc_0m	Number of times log in online banking with pc dec-19
tablet_0m	Number of times log in online banking with tablet dec-19
national_transfers_0m	National transfers dec-19
transfers_0m	Total transfers dec-19
app_par_0m	Number of times log in app dec-19
web_par_0m	Number of times log in web dec-19
amount_cards_0m	Total number of debit cards dec-19
amount_trfr_0m	Total amount of transfers dec-19
amount_salarypenr_0m	Salary dec-19
amount_rec_0m	Amount of domiciliated receipts(€)
amount_total_cards_0m	Total number of cards dec-19
num_trfr_0m	Total transfers inside Spain dec-19
num_trfe_0m	Total transfers outside Spain dec-19
num_rec_0m	Number of domiciliated receipts(€)
num_card_postpone_payment_3t4t_delta	Number of card postpone payments
aved_amount_postpone_payment_3t4t_delta	average amount of card postpone payments
previous_loans	Boolean previous loans (Y,N)
months_since_loan	Number of months since last loan
car_years	Number of years of your card
House_years	Years since the house was build
got_loan	Boolean Got loan

Process the data for analysis and pre-modeling

The client files were extracted with SAS to a CSV. The original information from the company's DDBB was later anonymized. Population and other information related to customer location was taken from the INE webside.

DataFrame (notice this dataframe uses the ";" delimeter). Chunksize was also used to upload the file.

Path is used to return a normalized absolutized version of the pathname.

There were originally 400k instances and 51 features. Non residents were filtered and deleted to avoid noise in the analysys.

The dataframe used contains 395.937 instances and 44 features. 388.932 customers without a personal loan and 7.005 with a personal loan are used as the dependent variable.

21 of the 44 features have, at least, one null value and non duplicates were found.

Categorical and numerical features were treated separatelly. "sample" used for every iteration to make sure it is done correctly.

1. Categorical feature preparation

- Create a Dataframe "categorical_file" only with categorical features.
- Fill in NaN accordingly to prepare to encode categorical features.
- Replace special values of these categorical features.
- Change all categorical variables into dummies(get_dummies).

1.2 Numerical feature preparation

- Recuperate the numerical features previouslly dropped. Create a dataframe
- Replace commas representing decimal places with periodos for all columns with 'amount' as part of the name.
- Replace columns with Nan with 0.
- Change object dtype to float.

1.3 Merge categorical and numerical datasets

- Merge both files, dropping feature 'got loan' from one of them.

- Use POP to remove the item at the given position in the list. Avoid feature duplication.
- Change all column types to numerics and downcast them.
- Basic Dataframe statistics.

Process data for visualization

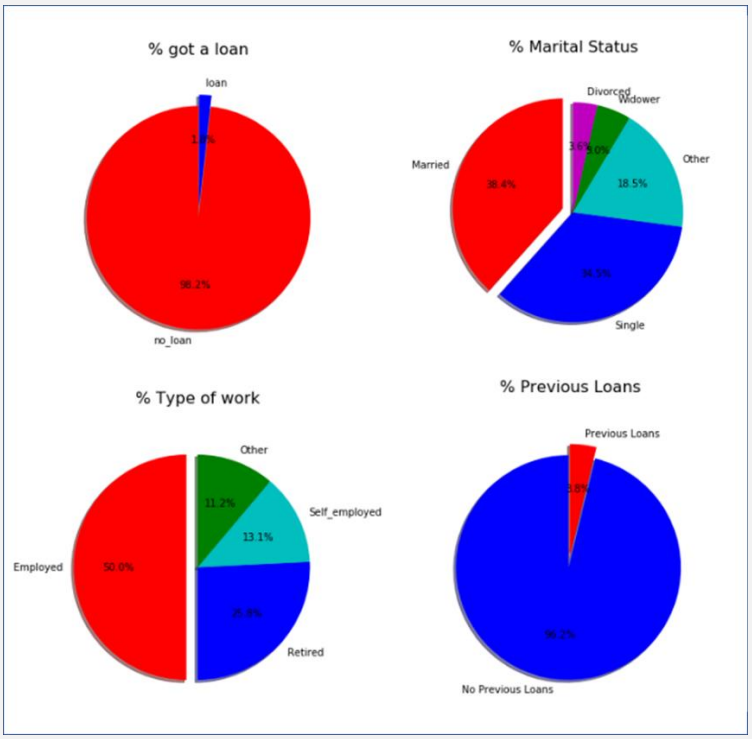
2. Data for visualization

Different feature treatment followed for visualization than for pre-modeling

2.1 Categorical features visualization

- Changed all feature dtype to object.
- Changed all but score categorical variables into dummies using a loop. Merged all new columns into one DF (label encoding maintains a column, labeling the categorical values as numbers).
- Ploted all the categorical features without differenciating loan or not loan(pie plot/subplot)(graphic 1).

Graphic 1



All the categorical feature plots can be found in the notebook

Insights:

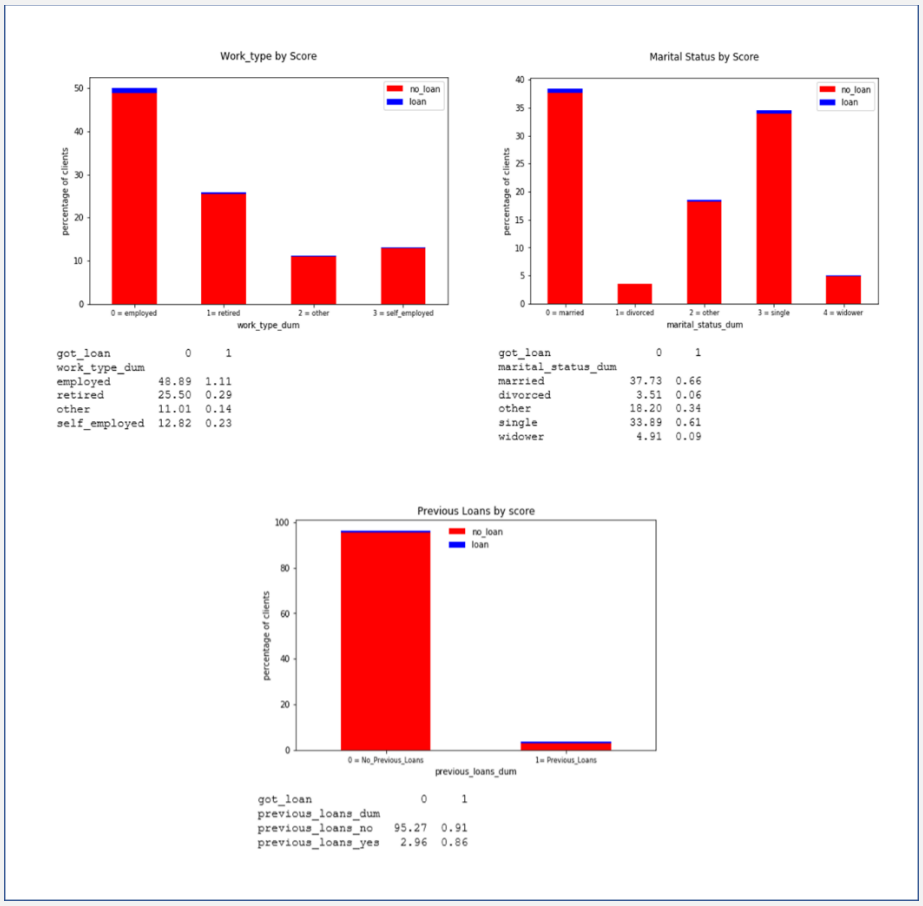
- 1.8% of the population studied have contracted a personal loan.
- 54.3% of the population studied have levesl of studies 1 or 2 (no studies or up to basic studies).
- 50% employed and 25% retired.
- 38.4% married and 34.5% single.
- 71.5% belong to mass market segment.
- 14.7% have current mortgages.
- 3.8% have had previous personal loans.

2.1.1 Categorical features visualization (bar plot)

Same analysis than before but differentiating population with and without a personal loan (Graphic 2).

- Stacked bar graphs.
- Created a new column counting the clients for each group.
- Created a percentage for each group category.
- Dropped count to keep only the percentage column.
- Added the percentage column.
- Transposed the df to create the stacked bar plot.
- Transposed to show the detail table underneath and renamed the legend.
- Rounded the values in the table under bar plot.

Graphic 2



All the categorical feature plots can be found in the notebook

Insights:

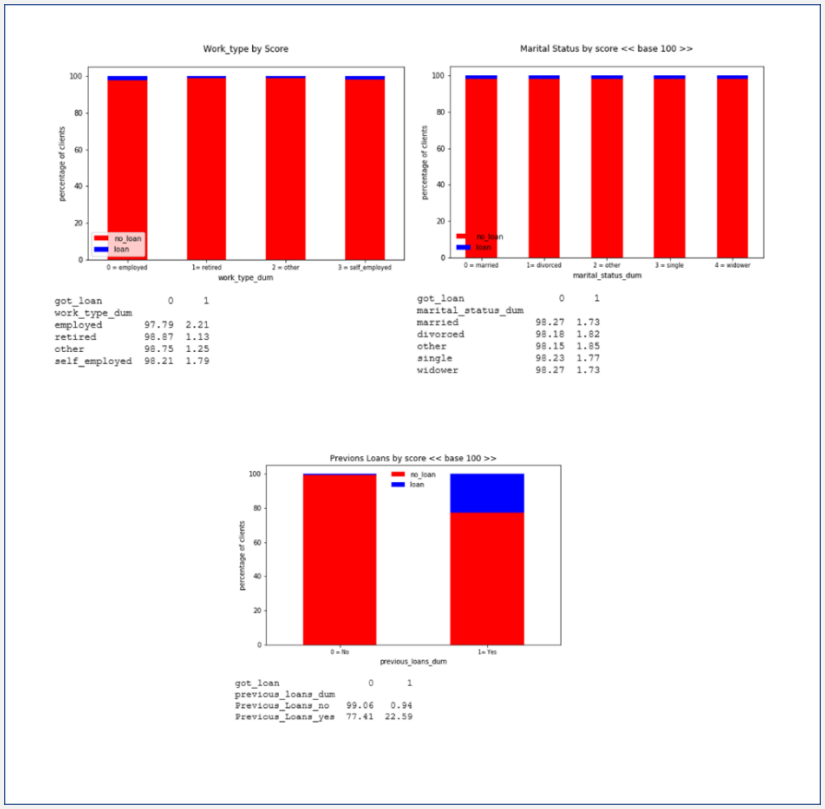
- Out of the total population, 1.08% were men and contracted a loan vs 0.68% women.
- Out of the total population, 0.70% have study level 2 and contracted a loan followed by study level 3 with 0.4% (study level 1- 0.31%).
- Out of the total population, 1.11% are population employed and contracted a loan followed by retired 0.29%.
- Out of the total population, 0.66% are married and contracted a loan followed by 0.61% single.
- Out of the total population, 1.37% belong to the mass market segment and contracted a loan followed by self-employed with 0.24%.
- 1.45% with no mortage contracted a personal loan vs 0.32% with mortgage.
- 0.91% with previous personal loans contracted a loan vs 0.86% without previous personal loans.

2.1.2 Categorical features visualization (base 100)

Same as the previos analysis, but taking into account each population weight within the sample.

Each population with and without a loan was analysed to know the percentage in base 100 of each group (Graphic 3).

Graphic 3



Insights:

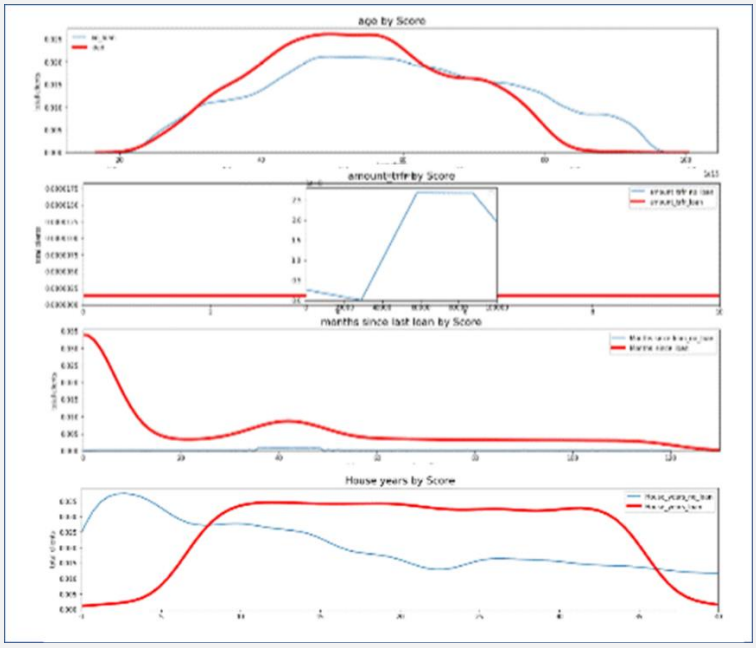
- 1.86% of total men contracted a loan vs. 1.65% women.
- 2.16% of the population with study level 2, contracted a loan followed by study level 6 with 1.84% (study level 1 - 1.43%).
- 2.21% of the population employed contracted a loan followed by self-employed 1.79%.
- 1.82% of divorced population contracted a loan followed by 1.77% singles.
- 1.98% of the population belong to the self-employed segment contracted a loan followed by mass market with 1.92%.
- 2.19% with a mortage, contracted a personal loan vs. 1.70% without a mortgage .
- 22.59% with a previous personal loan contracted a loan vs. 0.94% without a previous personal loan.

2.2 Numercial visualization

Kdeplot from *seaborn* will be used. Kernel Density Estimate was used for visualizing of the Probability Density of a continuous variable.

- Set the title and size.
- Defined the feature to plot together with got loan values 0 and 1.Defined different color and width for got_loan=1 .
- Inserted graph for 'aveg_amount_invest_funds_0m' and 'amount_trfr_0m' features to better distinguish values 0 and 1.
- Set the frame and the legend positioning.
- Names the plot.

Graphic 4



Insights:

- It seems that between 30 and 65, there is a higher probability to contract a personal loan.
- Up to 20 months since your last lown paymet and between 30 - 60 months after, there is a higher probability to contract a new personal loan.
- When the house is between 10-35 years old, the probability of contracting a personal loan increases.
- The probability of getting a personal loan is higher if the person spends between 100€ and 1.500€ on his/her ccard.
- Clients with no significant expenses on their personal credit cards, have more probability of contracting a personal loan. (Most likely due to risk aversion from lenders).
- Clients with more than 30k euros in funds have a higher probability of contracting a loan (tend to happen to avoid customer's descapitalization).

- Clients with 8-year-old vehicles have the highest probabiliy of contracting a personal loan to purchase a new car.
- Digital clients who check bank details through their Bank App have a higher probability of contrating a loan (most likely due to the online offer for these types of loans).

2.3 Customers Geolocalization

The objective is to see if there is a concentration either of total clients or clients who contracted a loan in a specific area (Graphic 5).

'Folium.Map' and 'Folium.Choropleth' were used to define 2 separates groups and folium 'LayerControl' used to add them into the same html graph.

- Downloaded the map 4 files to local folder and read de 'shp' file.
- Selected the column, with a significant value, from 'my file,change' column name to be used as key with the previous file 'polygon_pc' and grouped the clients by postal code.
- The map showed 2 separate groups. Total clients and clients who contracted loans.
- Weighed no loan clients by province.
- 2 separate groups were plotted. Choropleth is used for binding the data between Pandas DataFrames and JSON geometries. Sequential color schemes are built-into the library and can be passed to visualize different combinations. Creates a LayerControl object to be added on a folium map.
- Saved Info in html file.

Graphic 5

Clients with loan by Province

Weighted Clients with Personal Loan by Province

Insight:

- Comparing to the total number of clients in the population, the results obatained show that there are more clients from the Cataluña and Malaga areas.
- However, the success rate is the highest in the Northwest and Cataluña áreas (as shown in the previous graphic).

[1]Folium: https://python-visualization.github.io/folium/quickstart.html

Feature Selection and Outliers

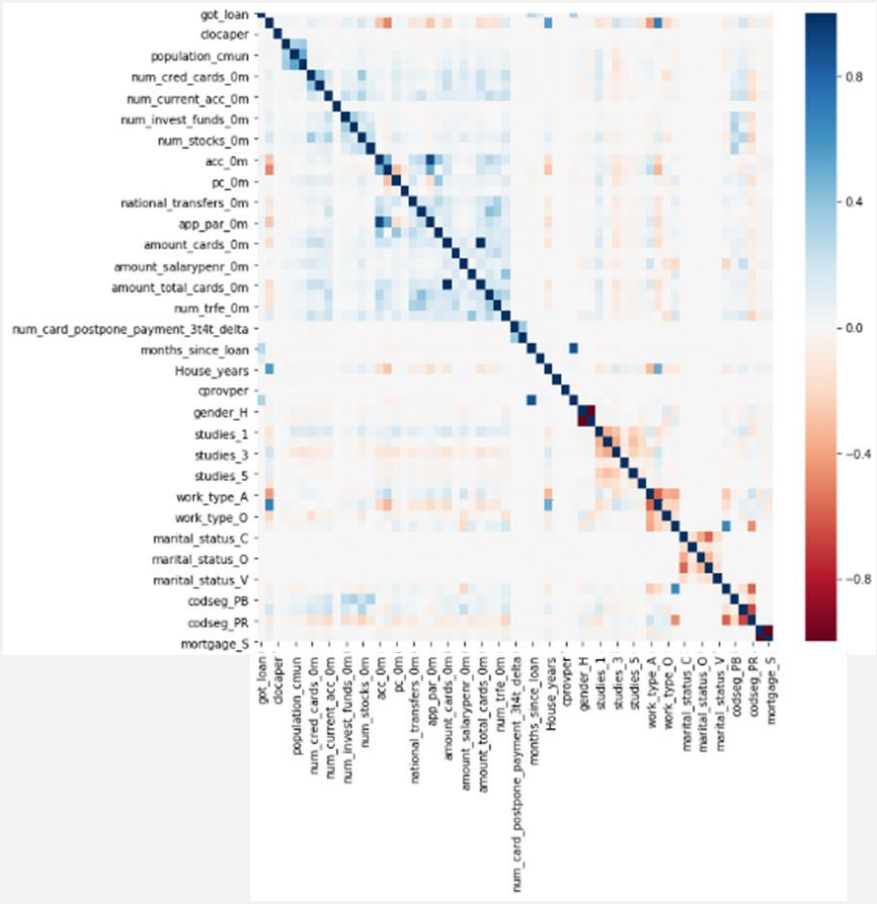
3. Feature Selection

Four different correlation and feature selection analysis have been defining the "final" features used.

3.1 Heatmap

A heatmap is very useful in visualizing the concentration of values between two dimensions of the dataset and the correlation between the two.

Graphic 6



Insights:

- A first contact revealed how some features are highly correlated with one other. Some of them also more correlated than others with the dependent variable. See further detail in 2.2.

3.2 Top varible absolute correlation

The following of the highest absolute correlation pairs from a correlation matrix:

- Unstack and order to get the most correlated pairs.
- Set_option allows displaying all instances.
- Drop all duplicates and ascending order to display the more correlated only.
- Eliminate variables that are highly correlated with others.

```
Top absolute Correlation pairs

Out[123]: amount_total_cards_0m  amount_cards_0m       1.0000
          mortgage_S             mortgage_S            1.0000
          mortgage_N             mortgage_S            0.9989
          acc_0m                 app_par_0m            0.9088
          previous_loans         months_since_loan     0.8703
          work_type_J            age                   0.6861
          codseg_PI              codseg_PR             0.6683
          work_type_P            codseg_CN             0.6549
          work_type_A            work_type_J           0.5895
          codseg_PR              codseg_CN             0.5847
          marital_status_C       marital_status_S      0.5729
          House_years            age                   0.5660
          disporent_cmun         population_cmun       0.5182
          app_par_0m             mobile_0m             0.5151
          House_years            work_type_J           0.4966
          dtype: float64
```

Insights:

- Useful to avoid modelling problems by eliminating varibles that are highly correlated with others.For this exercise, > 0.7 is going to be considered high correlation.

3.3 Top correlated features with the dependent variable

- The highest correlated variable is ‘previous_loans’.
- The features to be later included in the model will be plotted to clearly see the correlation between them (Pairplot).
- As previously seen ‘months_since_loan’ and ‘previous_loans’ are highly correlated with one other.

```
Out[40]: got_loan              1.000000
         months_since_loan     0.278317
         previous_loans        0.314747
         Name: got_loan, dtype: float64
```

Insights:

- There are no highly correlated variables with the output variable ‘got_Loan’. However, correlation does not mean causality so continued analysis of the variables was required.

3.4 Backward feature elimination [2]

All features were included in this model. Afterwards, the performance of the model was verified. The worst performing features were removed iterativilly until the performance of the model was acceptable. The performance metric used was 'pvalue'.

- Created loop with *while* to build the model in every new iteration
- Adding constant column of ones, mandatory for sm.OLS model.
- Features with pvalues > 0.05 were elimintated.

The final features number was: 33

```
features_to_model = ['got_loan',
                     'age',
                     'population_cmun',
                     'num_cred_cards_0m', 'aveg_amount_cred_card_0m', 'amount_cards_0m', 'amount_total_cards_0m',
                     'num_stocks_0m',
                     'acc_0m', 'mobile_0m', 'pc_0m', 'tablet_0m', 'app_par_0m', 'web_par_0m',
                     'amount_rec_0m', 'num_rec_0m',
                     'aved_amount_postpone_payment_3t4t_delta',
                     'months_since_loan',
                     'car_years',
                     'House_years',
                     'cprovper',
                     'previous_loans',
                     'gender_V',
                     'studies_1', 'studies_2', 'studies_3',
                     'work_type_A', 'work_type_J', 'work_type_O', 'work_type_P',
                     'codseg_PB', 'codseg_PI', 'mortgage_N']
```

Insights:

- Since the final work explains why a specific customer is selected by the model and it is also known that <<Global features important>> may not be important in the local context, the features selected previously (Backward Elimination) were used to prove this and to give more information to sales representatives.

4. Outliers [3]

2 methods to study outliers were carried out.(IQR & Zscore)

4.1 IQR (mean)

<< The interquartile range (IQR) is the difference between the 75th and 25th percentile of the data. It is a measure of the dispersion similar to standard deviation or variance, but is much more robust against outliers>> *SciPy.org*

<<The **first quartile (Q1)**, is defined as the middle number between the smallest number and the median of the data set, the **second quartile (Q2)** – **median** of the given data set while the **third quartile (Q3)**, is the middle number between the median and the largest value of the data set. **IQR = Q3 – Q1** covers the center of the distribution and contains 50% of the observations>>
Geeksforgeeks

Followed process:

- Drop features factorized.
- First and third quartile were defined.
- 1.5 times the IQR is a suspected outlier and 3 times the IQR above or bellow the Q1 or Q3 accodingly is considered a definitive outlier.
- A dataset was created with the instances with outliers in any feature.
- Dropped columns, included in the outliers´ review dataset.
- Both dataset were concated to get all the columns back without outliers.

```
Dataset with outliers (395937, 61)
Dataset without outliers (163467, 61)
```

Insigths:

- IQR does not work for outliers in this particular dataset due to the high data dispersion.Relevant Info would be lost for the model if used.

4.2 zScore (Standard deviation)

zscore tells us how many standard deviations away a value is from the mean.

- If the zscore is > than 3, that point can be classified as an outlier. Any point +/- 3 standard deviations would be an outlier.
- Features not factorized previously used.
- Created a dataset with the instances with outliers in any featured.
- Dropped columns included in the outliers´ review dataset.
- Concated both dataset to get all the columns back without outliers

```
Dataset with outliers (395937, 61)
Dataset without outliers (371986, 61)
```

Insigths:

- zscore is a better solution than IQR for this dataset, however has not been used for the model due to the fact that most sophisticated models used are prepared to deal with outliers. The tests that have been run, do not improve the results without outliers.

[2]Backward elimination: https://towardsdatascience.com/p-value-basics-with-python-code-ae5316197c52
[3]Outliers: https://medium.com/datadriveninvestor/finding-outliers-in-dataset-using-python-efc3fce6ce32

Model approach and Model

5. Model approach

4 different models have been tested for this analysis plus a final stacking ensemble method which will include the four of them:

- Logistic regression.
- Decision tree classifier.
- Random forest.
- XGBBoost.

Every model will be tested for:

- Normalized data.
- Normalized-balanced data.
- Non normalized data.
- Non normalized-balanced data.

5.1 Dataset Normalization

- Apply list comprenhension.
- This is column-wise and min-max normalization.
- Could also be used mean & std normalization method with Pandas.
- Print the results to make sure it is correct.

```
non normalized: (395937, 31)
normalized: (395937, 31)
non normalized 1: (7005, 31)
non normalized 0: (388932, 31)
normalized 1: (7005, 31)
normalized 0: (388932, 31)
```

Insights:

- Since only 1.8% of the clients contracted a loan ,checking the performance of the model using also balanced data is required.

Balancing Dataset

Unbalanced dataset

- Sklearn used for 'train' and 'test' split for the 90% of non normalized and normalized dataset.
- The function `train_test_split` made random partitions for the two subsets below. A random state and a 0.25 test_size are also specified.
- Printed results fro checking purposed.

Non normalized

```
X: (356343, 32)
X_train: (267257, 32)
X_test: (89086, 32)
y_train: (267257, 1)
y_test: (89086, 1)
n° of 1  (1540, 1)
n° of 0  (87546, 1)
```

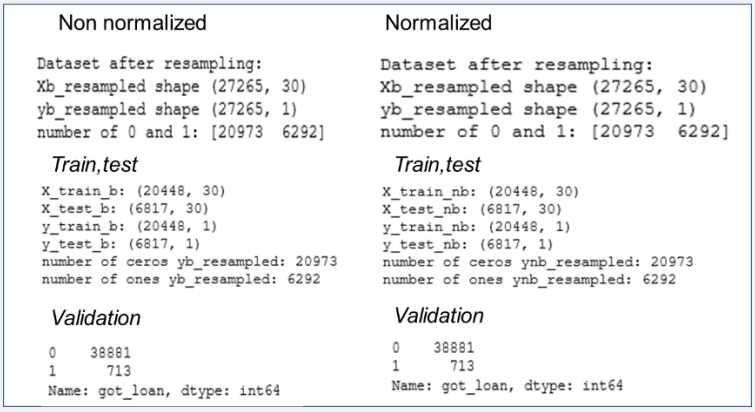
Non normalized

```
X_train_n: (267257, 32)
X_test_n: (89086, 32)
y_train_n: (267257, 1)
y_test_n: (89086, 1)
0.0    87546
1.0     1540
Name: got_loan, dtype: int64
```

Balanced dataset

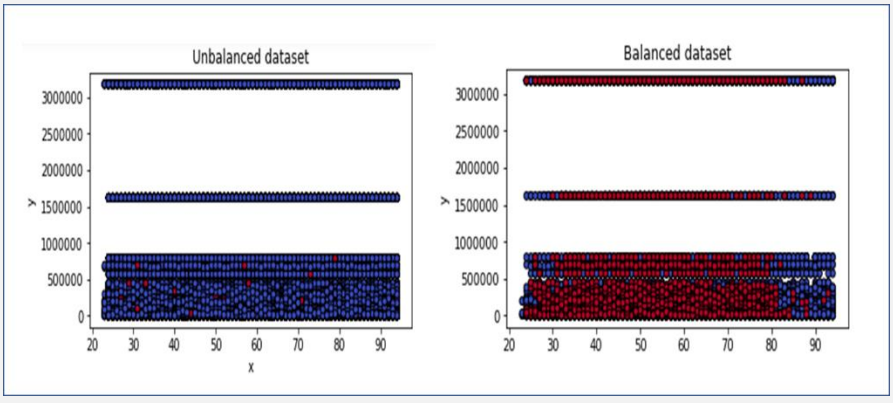
- Repited the same for normalized and non normalized dataset but now balancing the dataset.

- Defined the features and the target.
- Synthetic Minority Oversampling Technique(SMOTENC) used for Over-sampling, under-sampling and the combination between both.
- Tested with the models, the one with best results is under-sampling.
- Defined 30% under- sampling strategy.
- Printed resutls for checking purposes.



5.2 Visualization unbalanced and balanced dataset

- Converted to an array with numpy.
- Converted to float/int to be able to use parameters like cmap when plotted.
- marker='o'.c=y_arr_list, to higlight the dependent variable, points size s=25,
- edgecolor='k', edge color of the marker,cmap=plt.cm.coolwarm



Train,Test,Validation

- Hold out 10% for validation

- ‘sample’ used to make sure is random
- No replacement
- Print the results

```
90% of clients
non normalized 1: (6292, 31)
non normalized 0: (350051, 31)
normalized 1: (6292, 31)
normalized 0: (350051, 31)
```

```
10% of clients
non normalized val 1 (713, 31)
non normalized val 0: (38881, 31)
normalized val 1: (713, 31)
normalized val 0: (38881, 31)
```

6. Model

- 5 models were used to decide the one which better adjusted to the binary classification analysis (Logistic Regression[4], Decision Tree Classifier[5], Random Forest Classifier[6], XGBoost[7], Stack ensemble model[8]).
- All models were run with the 4 type of pre-processed datasets.
- The approaches were the same for all 4:
  - Creating the model.
  - Feeding the training data into the model to fit.
  - Tuning its internal parameters to maximize predictions.
  - Testing.
  - Checking performance with the following evaluation matrix: confusion_matrix, acc score, ROC curve. For the ROC curve some transformations were required to avoid scalar issues.
  - Getting a classification report for lts for all the models and iterations. ROC results have been prioritized as well as hyperparameters built to reach maximum of true positives.
  - Using the model for the validation dataset(10%) with the same evaluation matrix for validation purposes.

Metrics:

Models		Logistic regression	Decision tree Classifier	Random Forest	XGBoost
Non normalized balanced	Test	F1-score(0): 0.76 F1-score(1): 0.39 AUC Score: 0.65 ROC: 0.63	F1-score(0): 0.90 F1-score(1): 0.73 AUC Score: 0.85 ROC: 0.89	F1-score(0): 0.94 F1-score(1): 0.79 AUC Score: 0.90 ROC: 0.96	F1-score(0): 0.94 F1-score(1): 0.79 AUC Score: 0.90 ROC: 0.96
	Validation	F1-score(0): 0.83 F1-score(1): 0.06 AUC Score: 0.71 ROC: 0.64	F1-score(0): 0.93 F1-score(1): 0.18 AUC Score: 0.86 ROC: 0.90	F1-score(0): 0.97 F1-score(1): 0.33 AUC Score: 0.94 ROC: 0.96	F1-score(0): 0.98 F1-score(1): 0.37 AUC Score: 0.96 ROC: 0.97
Normalized balanced	Test	F1-score(0): 0.85 F1-score(1): 0.64 AUC Score: 0.79 ROC: 0.89	F1-score(0): 0.91 F1-score(1): 0.75 AUC Score: 0.87 ROC: 0.90	F1-score(0): 0.94 F1-score(1): 0.78 AUC Score: 0.91 ROC: 0.88	F1-score(0): 0.94 F1-score(1): 0.79 AUC Score: 0.79 ROC: 0.97
	Validation	F1-score(0): 0.89 F1-score(1): 0.12 AUC Score: 0.80 ROC: 0.88	F1-score(0): 0.93 F1-score(1): 0.19 AUC Score: 0.87 ROC: 0.90	F1-score(0): 0.94 F1-score(1): 0.37 AUC Score: 0.95 ROC: 0.86	F1-score(0): 0.98 F1-score(1): 0.39 AUC Score: 0.96 ROC: 0.97

Stacking ensemble method

The Stacking ensemble technique was used to increase the predictive force of the classifier. The stability of the final model is increased. Even when they are individually give back bad results.

<<The main principle behind ensemble modelling is to group weak learners together to form one strong learner>>Robert R.F. DeFilippi

After importing the libraries required, the list of classifier models previoully used were included.

Metrics:

Models		Stacked model
Non normalized balanced	Test	F1-score(0): 0.94 F1-score(1): 0.78 AUC Score: 0.90 ROC: 0.92
	Validation	F1-score(0):0.97 F1-score(1): 0.32 AUC Score: 0.94 ROC: 0.85
Normalized balanced	Test	F1-score(0): 0.94 F1-score(1): 0.79 AUC Score: 0.90 ROC: 0.89
	Validation	F1-score(0): 0.97 F1-score(1): 0.33 AUC Score: 0.94 ROC: 0.90

ie: Balanced test & validation results (Stacking ensemble model)

Stacking roc-auc

```
3-fold cross validation:

roc_auc: 0.89 (+/- 0.00) [lg]
roc_auc: 0.90 (+/- 0.00) [dt]
roc_auc: 0.96 (+/- 0.00) [rf]
roc_auc: 0.88 (+/- 0.03) [StackingClassifier]
```

Stacking testing

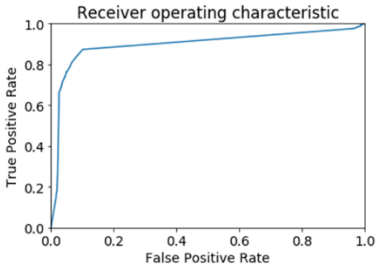
```
=== Confusion Matrix ===
[[4968  256]
 [ 381 1212]]
=== classification_report ===
              precision    recall  f1-score   support

         0.0       0.93      0.95      0.94      5224
         1.0       0.83      0.76      0.79      1593

    accuracy                           0.91      6817
   macro avg       0.88      0.86      0.87      6817
weighted avg       0.90      0.91      0.91      6817

=== accuracy_score ===
0.9065571365703389
```

```
=== roc_curve ===
0.8966439120616708
```



Stacking validation

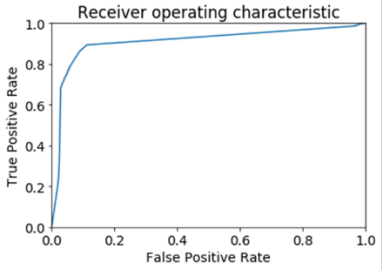
```
=== classification_report ===
              precision    recall  f1-score   support

         0.0       1.00      0.95      0.97     38881
         1.0       0.21      0.77      0.33       713

    accuracy                           0.94     39594
   macro avg       0.60      0.86      0.65     39594
weighted avg       0.98      0.94      0.96     39594

=== accuracy_score ===
0.9427438500782946
```

```
=== roc_curve ===
0.9090131094796281
```



Insights:

- The classification report for unbalanced dataset showed overfitting on the value 0. In this particular case, balancing data improved the results.

- For fine tuning the models, ‘randint’ was used. The ‘randint’ module would assist to initialize random values for the hyper parameters specified.

- For simplification purposes, only balanced results were shown. All the results can be found in previous iterations of the notebook.

- "The problem is that a single metric, such as classification accuracy, is an incomplete description of most real-world tasks." (Doshi-Velez and Kim 2017) *christophm.github.io*

- For this particular exercise, the ROC has been prioratized to reach the maximum amount of real potential clients (this illustrates how accurate the model is to detect true positives by ploting true positives againts false positives).

[4][5]Logistic regression: : https://christophm.github.io/interpretable-ml-book/storytime.html

[6]Random forest: https://jakevdp.github.io/PythonDataScienceHandbook/05.08-random-forests.html

[7]XGBoost: https://towardsdatascience.com/xgboost-python-example-42777d01001e

[8]Stacking ensemble: https://medium.com/@rrfd/boosting-bagging-and-stacking-ensemble-methods-with-sklearn-and-mlens-a455c0c982de

Business Plan –Why and How

7. Business Plan

A Business plan has been created to maximize, not only success rate but also to reduce costs.

With the Business Plan, the two main questions of this work will be answered.

- *Why the "black box"model is predicting a client over another?*
- *How can the results help build rapport with the client?*

<< Sales targets and leads can proiferate but they are of no use if they just create a morass of information for the sales person to go through without insights which makes Sales reps job more complicated>> Big Data ,Analytics, and the Future of Marketing and Sales[9]

To be able to carry out the analysis, I am going to analyse the predictions based on:

- Communication channels.
- Cost per contact.
- Percentage of customers who open & read the communication/message.
- Individual Important Features as oppossed to Models Global Feature Importance. Approaching customers specially from the Sales reprsentatives perspective depends on the local feature important selection.

Since new customers are not available to be used for the model, the validation file customers results will be used as an extrapolation to 100k customers.

As a result of these calculations, only 21,9% of the customers identified by the model with more than 60% probability (21.900 clients) will be approached.

7.1 Initial approach and basic information *(Why is important, from the business perspective, to avoid getting lost in translation with Sales representatives)*

Basic assumptions are created to be able to build a Business case.

Assumptions		
average loan	6.000 €	
margen per Loan	4%	240 €
loan term	4 years	
total number of clients	100k	
Success prediced cust	22%	21.900

Channels Basic information		
Channels	Unit cost	% open the communication
web	1 €	5%
email	3 €	3%
whatsApp	5 €	15%
mail	15 €	20%
phone	30 €	35%
mail + phone	45 €	55%
branch	100 €	70%
email + branch	103 €	73%
mail + branch	115 €	90%

To Break even, a mínimum number of customers is required based on the opening and success rates).Previous cost information was also used.

Minimum number of clientes by Open & Success rate								
Success Rate	Web	email	whatsApp	mail	phone	mail + phone	branch	email + branch
25%	80	133	27	20	11	7	5	4
35%	57	95	19	14	8	5	4	3
45%	44	74	15	11	6	4	3	2
55%	36	61	12	9	5	3	2	2
65%	31	51	10	8	4	3	2	2
75%	27	44	9	7	4	2	2	1
85%	24	39	8	6	3	2	2	1
95%	21	35	7	5	3	2	1	1
100%	20	33	7	5	3	2	1	1

Taking the previous analysis results to our dataset, the following table shows us the best channel approach that should be used in order to maximize our results:

	Unit cost	1	3	5	15	30	45	100	103	115	
	% open rate	5%	3%	15%	20%	35%	55%	70%	73%	90%	
Maximun profitability matrix (based on Unit Cost,Success Rate & Open Rate)											
Success Rate	number of clients	Web	email	whatsApp	mail	phone	mail + phone	branch	email + branch	mail + branch	Most profitable Channel
25%	40	81	-48	161	-121	-363	-484	-2.339	-2.388	-2.460	whatsApp
33%	40	121	-24	282	40	-81	-40	-1.775	-1.799	-1.734	whatsApp
43%	121	501	10	1.262	674	726	1.400	-3.388	-3.377	-2.714	mail + phone
50%	484	2.420	290	6.292	4.356	5.808	10.164	-7.744	-7.453	-3.388	mail + phone
55%	444	2.484	426	6.566	5.058	7.187	12.245	-3.372	-2.946	1.686	mail + phone
57%	323	1.890	360	5.024	4.010	5.808	9.818	-1.291	-931	2.719	mail + phone
60%	1.089	6.751	1.437	18.077	15.028	22.215	37.242	871	2.309	15.899	mail + phone
61%	444	2.810	621	7.542	6.359	9.464	15.823	1.183	1.804	7.542	mail + phone
62%	645	4.120	923	11.069	9.382	13.998	23.380	2.184	3.107	11.566	mail + phone
64%	282	1.874	447	5.056	4.389	6.622	11.010	1.951	2.397	6.339	mail + phone
64%	363	2.437	591	6.586	5.756	8.712	14.467	2.904	3.495	8.660	mail + phone
65%	444	3.001	736	8.116	7.124	10.804	17.929	3.862	4.598	10.987	mail + phone
67%	1.129	7.905	2.033	21.456	19.198	29.361	48.559	13.551	15.584	32.749	mail + phone
69%	444	3.216	865	8.762	7.986	12.311	20.297	6.877	7.742	14.862	mail + phone
69%	363	2.653	720	7.232	6.617	10.219	16.837	5.919	6.640	12.537	mail + phone
70%	282	2.089	576	5.703	5.251	8.131	13.382	4.969	5.545	10.220	mail + phone
71%	605	4.581	1.296	12.532	11.667	18.149	29.816	12.099	13.396	23.767	mail + phone
72%	1.049	8.039	2.307	22.021	20.623	32.158	52.780	22.371	24.677	42.993	mail + phone
75%	726	5.808	1.742	15.971	15.245	23.957	39.202	18.875	20.617	34.120	mail + phone
76%	645	5.255	1.604	14.473	13.920	21.940	35.860	18.069	19.673	31.989	mail + phone
77%	403	3.320	1.024	9.152	8.842	13.961	22.803	11.789	12.813	20.631	mail + phone
78%	282	2.353	734	6.493	6.305	9.975	16.280	8.658	9.392	14.963	mail + phone
79%	1.210	10.253	3.248	28.338	27.701	43.940	71.641	39.482	42.730	67.184	mail + phone
80%	645	5.550	1.781	15.358	15.100	24.005	39.105	22.198	23.979	37.299	mail + phone
85%	686	6.308	2.139	17.552	17.689	28.385	46.075	29.345	31.484	47.035	mail + branch
87%	524	4.929	1.699	13.737	13.947	22.440	36.387	23.909	25.607	37.855	mail + branch
88%	565	5.364	1.863	14.963	15.245	24.562	39.807	26.538	28.401	41.783	mail + branch
89%	323	3.119	1.097	8.712	8.927	14.412	23.338	15.917	17.015	24.844	mail + branch
90%	726	7.114	2.526	19.891	20.472	33.104	53.576	37.170	39.696	57.642	mail + branch
92%	484	4.877	1.765	13.663	14.184	23.008	37.192	26.656	28.421	40.840	mail + branch
93%	524	5.318	1.932	14.905	15.505	25.167	40.671	29.361	31.294	44.866	mail + branch
93%	565	5.759	2.100	16.149	16.826	27.329	44.155	32.072	34.172	48.898	mail + branch
97%	1.129	11.955	4.463	33.606	35.397	57.710	93.107	70.249	74.712	105.646	mail + branch
100%	3.872	42.590	16.262	120.027	127.770	209.078	336.849	263.284	279.546	391.054	mail + branch

In summary:

A potential outcome of 21% ROI (~350k €) per year could result if these customers are addressed:

Summary						
Channels	% Success Rate	number of clients	Profitability	*Income*	*Cost*	*~ ROI per year*
whatsApp	<40%	80	444 €	*847 €*	*403 €*	28%
mail + phone	40%-80%	12.422	560.043 €	*1.119.037 €*	*558.994 €*	25%
mail + branch	>80%	9.397	840.464 €	*1.921.146 €*	*1.080.682 €*	19%
Total		21.899	1.400.951 €	*3.041.030 €*	*1.640.079 €*	**21%**

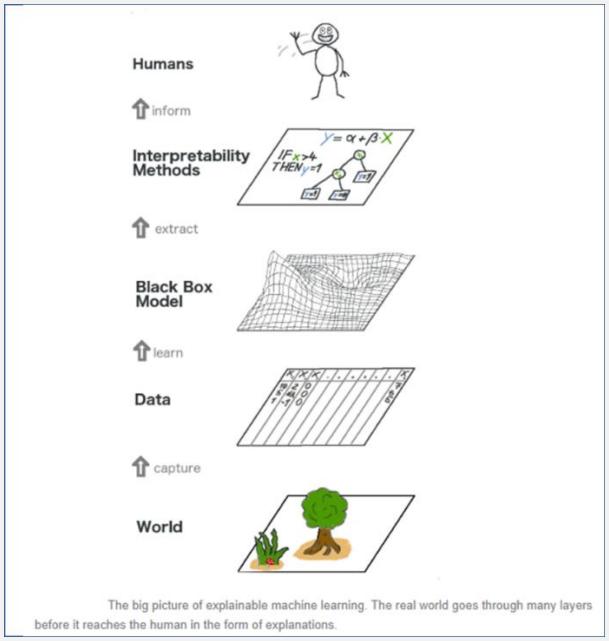
This Business Case has been developed by channel which would need to be completed with a comunication, marketing and product plan which is not included in this exercise. Further actions would be required and put into place(i.e triggers for customers for simulation purposes or through any other channel that provides further information). The following section will cover how to help sales representatives to approach the clients.

7.2 How can 'Lost in translation' be avoided with Sales representatives.

- After defining the number of customers (9.397) that will be managed by Sales Representative, explaining to them why this model has selected them and help them find the best manner to present it to the client is crucial.
- To do this comparing Global Feature Importance with Local Features selected for a client. For the Local Features selection, the model used was LIME(Local Interpretable Model-agnostic Explanations).

- LIME is a Python library for model explainability. LIME builds sparse linear models around each prediction to explain how the 'black box' model works in that local vicinity. According to <<Why should I Trust You?>> original paper[10], LIME is a subset of SHAP. Is used because is faster than Shap and it uses the Chi-Squared Test, a distribution-based approximation as a good approximation not as accurate as the Fisher Test used by Shap.[10]

- <<We capture the world by collecting data, and abstract it further by learning to predict the data (for the task) with a machine learning model. Interpretability is just another layer on top that helps humans understand>> Interpretable Machine Learning by Christoph Molnar [11]



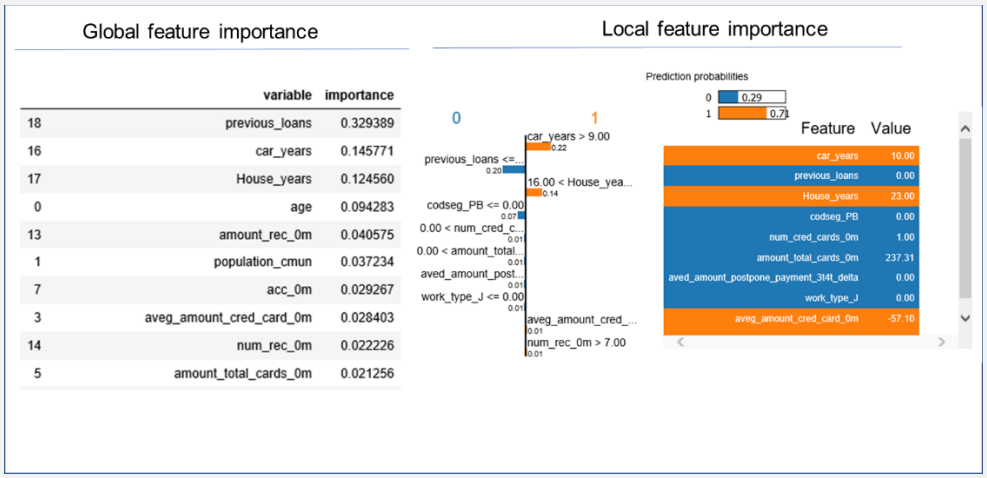
The big picture of explainable machine learning. The real world goes through many layers before it reaches the human in the form of explanations.

- A tool tradicionally used to detect bias in machine learning models, can also be used to give sales representatives tips on how best approaching a client. [12]

- Steps to follow(*Interpretable Machine Learning by Christoph Molnar*):

  - Select instance of interest for which you want to have an explanation of its black box prediction.
  - Perturb dataset and get the black box predictions for these new points.
  - Weight the new samples according to their proximity to the instance of interest.
  - Train a weighted, interpretable model on the dataset with the variations.
  - Explain the prediction by interpreting the local model.
  - Plotting results

Random Forest Classifier

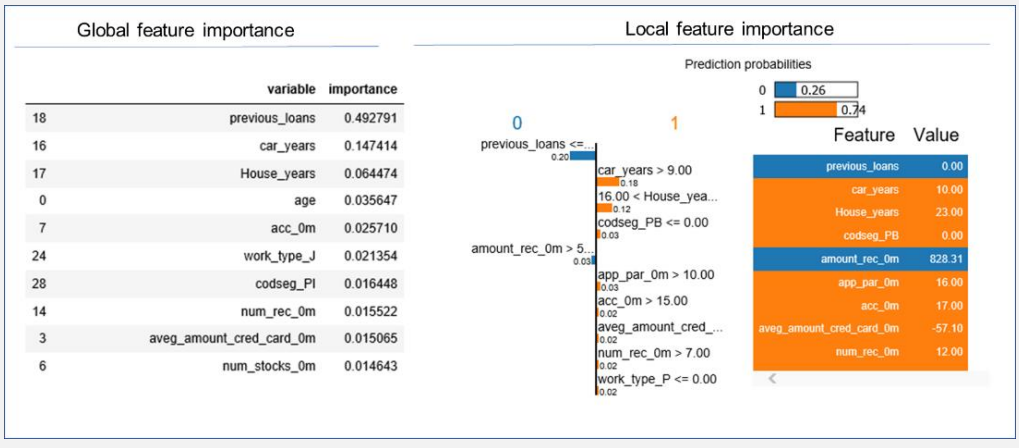


	variable	importance
18	previous_loans	0.329389
16	car_years	0.145771
17	House_years	0.124560
0	age	0.094283
13	amount_rec_0m	0.040575
1	population_cmun	0.037234
7	acc_0m	0.029267
3	aveg_amount_cred_card_0m	0.028403
14	num_rec_0m	0.022226
5	amount_total_cards_0m	0.021256

Insights:

- The model rates all the features but does not give you how each affects the prediction. By using LIME,one can see that this customer has a 10-years-old car and his house is 23. Sales Representatives can use this information to offer clients the possibility to change their car without having to pay all upfront or to ask if hi/she is thinking on reforming any part of his/her house.

XGBoost

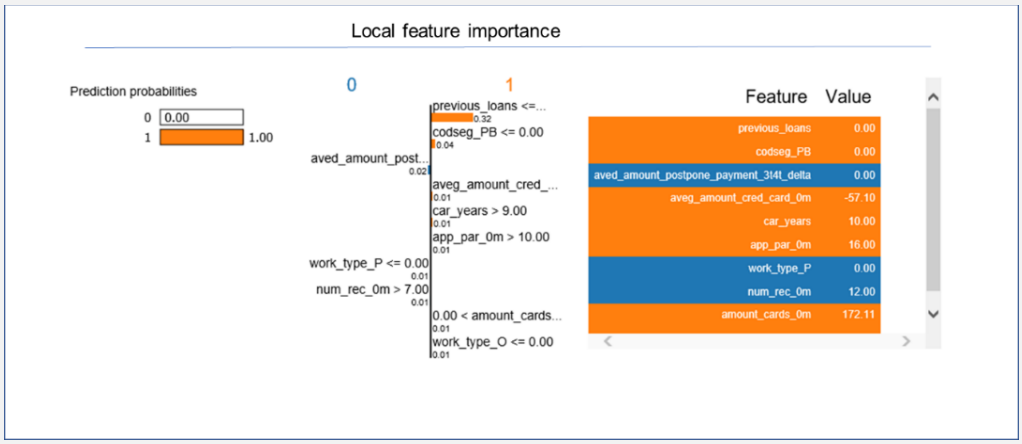


	variable	importance
18	previous_loans	0.492791
16	car_years	0.147414
17	House_years	0.064474
0	age	0.035647
7	acc_0m	0.025710
24	work_type_J	0.021354
28	codseg_PI	0.016448
14	num_rec_0m	0.015522
3	aveg_amount_cred_card_0m	0.015065
6	num_stocks_0m	0.014643

Insights:

- Similar results. As seen above, this model includes more features than the Random Forest Model. Features like no belonging to a private banking segment for instance.

Stacking model



Insights:

- The Global feature importance is the average of the global features of the trees used.
- The same results were obtained with the Stacking Method than with the previous models. It is interesting to see how this model has selected additional features, like how many times he/she checks her bank details through the app or the amount of money expent on cards.

[9]Big Data ,Analytics, and the Future of Marketing and Sales by Mckinsey Chef Marketing & Sales Officer Forum.
[10]Why should I Trust You? Explaining the Predictions of Any Classifier by Marco Tulio Ribeiro
[11]Interpretable Machine Learning: https://christophm.github.io/interpretable-ml-book/storytime.html
[12]LIME: https://blog.dominodatalab.com/shap-lime-python-libraries-part-1-great-explainers-pros-cons/

Important References:

[1]Folium: https://python-visualization.github.io/folium/quickstart.html

[2]Backward elimination: https://towardsdatascience.com/p-value-basics-with-python-code-ae5316197c52

[3]Outliers: https://medium.com/datadriveninvestor/finding-outliers-in-dataset-using-python-efc3fce6ce32

[4][5]Logistic regression: : https://christophm.github.io/interpretable-ml-book/storytime.html

[6]Random forest: https://jakevdp.github.io/PythonDataScienceHandbook/05.08-random-forests.html

[7]XGBoost: https://towardsdatascience.com/xgboost-python-example-42777d01001e

[8]Stacking ensemble: https://medium.com/datadriveninvestor/finding-outliers-in-dataset-using-python-efc3fce6ce32

[9]Interpretable Machine Learning: https://christophm.github.io/interpretable-ml-book/storytime.html

[10]Big Data ,Analytics, and the Future of Marketing and Sales by Mckinsey Chef Marketing & Sales Officer Forum.

[11]Why should I Trust You? Explaining the Predictions of Any Classifier by Marco Tulio Ribeiro

[12]LIME: https://blog.dominodatalab.com/shap-lime-python-libraries-part-1-great-explainers-pros-cons/

Other readings performed:

Estadística básica para machine learning: https://blog.findemor.es/2017/12/machine-learning-introduccion-estadistica-basica-python/

Data Science for Business by Foster Provos tan Tom Fawcett

A Survey of Methods for Explaining Black Box Models: https://dl.acm.org/doi/fullHtml/10.1145/3236009