University of Orléans

Scoring Methods

Authors:
Mickaël WALTER (50%)
Loïc PALMA (50%)

Lecturer:
Christophe RAULT
Referent:
Christophe RAULT

April 23, 2020

Abstract

The purpose of this project covering the Scoring Methods course teached in Master ESA is to put in the various concepts discussed in class, particularly those relating on credit scoring. This project is sponsored by RCI Bank & Services (Renault Credit International), a 100% subsidiary of Renault SA, which specializes in car financing and the distribution of related services. RCI Bank & Services supports the strategic development of sales of the Renault-Nissan Alliance brands and helps to win and retain Alliance brand clients by providing financing and related services. More precisely, the main missions of RCI Bank & Services are the development of internal credit risk models (Basel III, stress test) as well as scoring and the methodology of provisioning (credit risk scores, Marketing, IFRS9). They are the one providing the dataset. Our project aim to construct and develop a credit score. We will have to use a logistic regression and two machine learning algorithms in order to benchmark them with each other. In order to compare our models, the Gini and 10/X indices will be used. Due to obvious confidentiality reasons, the dataset could not be shared, as it is RCI Bank & Services' data. However, the GitHub repository including all codes is public and can be found here.

Ce projet, qui couvre le cours sur les méthodes de scoring enseigné dans le Master ESA, a pour objectif d'intégrer les divers concepts discutés en classe, en particulier ceux relatifs à la notation de crédit. Ce projet est sponsorisé par RCI Bank & Services (Renault Crédit International), filiale à 100% de Renault SA, spécialisée dans le financement automobile et la distribution de services liés. RCI Bank & Services soutient le développement stratégique des ventes des marques de l'Alliance Renault-Nissan et aide à gagner et à fidéliser les clients des marques de l'Alliance en fournissant des financements et des services liés. Plus précisément, les principales missions de RCI Bank & Services sont le développement de modèles de risque de crédit internes (Bâle III, tests de résistance), ainsi que la notation et la méthodologie de provisionnement (scores de risque de crédit, Marketing, IFRS9). Ce sont eux qui fournissent le jeu de données. Notre projet vise à construire et à développer un pointage de crédit. Nous devrons utiliser une régression logistique et deux algorithmes d'apprentissage automatique pour pouvoir les comparer. Afin de comparer nos modèles, les indices de Gini et 10/X seront utilisés. Pour des raisons évidentes de confidentialité, le jeu de données n'a pas pu être partagé car il s'agit des données de RCI Bank & Services. Cependant, le repository GitHub incluant tous les codes est public et peut être trouvé ici.

Contents

Chapter 1

Project Presentation

1.1 Dataset

Figure 1.1: RCI Bank & Services

The dataset is provided by RCI Bank & Services. This latter is a **real** and **raw** dataset from their own database.
RCI Bank & Services stands for Renault Crédit International Bank & Services. It is a 100% subsidiary of Renault SAS, which specializes in car financing and the distribution of related services. RCI Bank & Services supports the strategic development of sales of the Renault-Nissan Alliance brands and helps to win and retain Alliance brand clints by providing financing and related services.
RCI Bank & Services is present in more than 36 countries in the world: 25 countries in Europe, 4 countries in Eurasia and so on.
9 brands are financed: Renault, Dacia, Renault, Samsung, Nissan, Infiniti, Datsun, Alpine, Mitsubishi Motors and Lada.

Figure 1.2: Brands of the Renault-Nissan Alliance

Some key figures:

- 3,500 employees
- 1.8 million new funding applications (+1.6% vs 2017)
- 44.4 billion average productive assets (+12% vs 2017)
- 42.9% of new Alliance registrations financed (+ 0.3% VS 2017)

1.2 Objectives

The main goal of our project is to construct and develop a credit score. One using a Logistic Regression and two other using Machine Learning algorithms.

1.3 OSEMN Process

OSEMN is an acronym that stands for **O**btain, **S**crub, **E**xplore, **M**odel, and i**N**terpret. This latter can be used as a blueprint for working on data problems using machine learning tools.
Before even thinking about a possible answer, several complementary studies need to be carried out to ensure that quality of data and to be consistent with statistical modeling.

Here is a list of a few things we need to focus on:

- handle missing data;
- handle duplicates;
- handle outliers;
- feature engineering;
- think about a way to discretize quantitative variables.

Indeed the most important steps given any Data Science project. **GIGO** (garbage in, garbage out) is a concept common to computer science and mathematics: the quality of output is determined by the quality of the input. Similarly, if incorrect data is input to a program, the output is unlikely to be informative. For example, outliers can introduce bias in the estimation of models. However, the discretization of the variables limit their impacts. Duplicates can influence the score during modeling and Machine Learning algorithms can not take into account missing values.

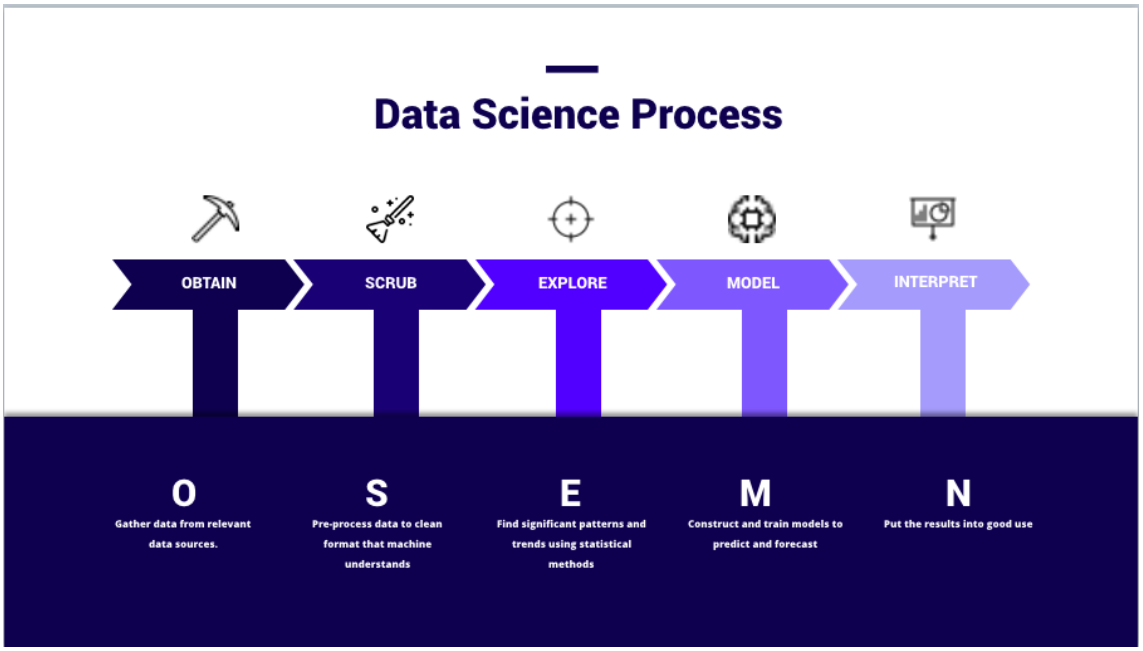

Figure 1.3: a Data Science Project Lifecycle

Note that in our case, the **O**btain step is already satisfied as RCI Bank & Services gave us the dataset.

Chapter 2

Data Cleansing

Data cleansing or **data cleaning** is the process of detecting and correcting (or removing) corrupt or inaccurate records from a record set, table, or database and refers to identifying incomplete, incorrect, inaccurate or irrelevant parts of the data and then replacing, modifying, or deleting the dirty or coarse data.

2.1 Variables

Our dataset's dimension is $(68538, 43)$. This means we do have 68538 observations that comes along with 42 predictors and our target variable.

2.1.1 Qualitative predictors

- **Type_pp**: Physical Person Vehicle Usage Code.
- **No_cnt_crypte**: Identifier of the encrypted contract.
- **No_par_crypte**: Identifier of the encrypted client.
- **genre_veh**: Type of vehicle.
- **produit**: Product type.
- **QUAL_VEH**: Quality of the vehicle.
- **IND_CLI_RNVA**: Renewing customer indicator.
- **EVPM_COPOT_PAI_GLB**: Evaluation of the payment behavior.
- **EVPA_PRTC**: Filing indicator for PROs.
- **COTE_BDF**: Rating of the Banque de France.
- **CSP**: Socio-professional class.
- **ETAT_CIVIL**: Civil status code.
- **MODE_HABI**: Code of living mode.
- **EVPM_COTE**: Evaluation of the quotation.
- **ind_fch_fcc**: FCC record indicator.
- **copot_**: Payment behavior.
- **pan_dir_**: PAN leader.
- **secteur_**: Activity area.
- **diag_fch_cli**: Filing indicator for PRIs.

2.1.2 Quantitative predictors

- **date_gest**: Month of the date of entry into management.
- **dt_dmd**: Date of demand.
- **DUREE**: Projected duration of financing.
- **MT_DMD**: Amount of funding.
- **PC_APPO**: Percent contribution.
- **AGE_VEH**: Age of the vehicle.
- **nb_imp_tot**: Number of outstanding payments.
- **nb_imp_an_0**: Number of outstanding payments from the last 12 months.
- **age**: Customer age.
- **mt_sal_men**: Amount monthly salary.
- **rev_men_autr**: Other monthly amount.
- **mt_alloc_men**: Amount of the monthly allowance.
- **nb_pers_chg**: Number of dependents.
- **MT_REV**: Amount of monthly income.
- **mt_loy_men_mena**: Amount of the monthly rent of the household.
- **mt_men_pre_immo**: Monthly amount of the mortgage.
- **mt_men_eng_mena**: Monthly amount various household commitments.
- **mt_charges**: Amount of charges.
- **MT_ECH**: Amount of the due date.
- **mt_ttc_veh**: Price of the vehicle.
- **part_loyer**: Share of maturity.
- **anc_emp**: Seniority of the job.
- **IND_IMP_REGU**: Number of outstanding payments settled over the last 12 months.

2.1.3 Target

- **WE18**[1]: Default indicator.

Meaning of colors

*Variables in **green** are variables available for both Professional and Private individuals.*
*Variables in **red** are variables available only for Private individuals.*
*Variables in **blue** are variables available only for Professional individuals.*

[1] WE18 is a dummy variable, being **1** if the individual is in default and **0** if he is not. One can be considered as default (1) if he was in default during the past 18 months.

2.2 Duplicate Data & Missing Values

Duplicate Data are entries that have been added by a system user multiple times. In statistics, **missing data**, or **missing values**, occur when no data value is stored for the variable in an observation. Missing data are a common occurrence and can have a significant effect on the conclusions that can be drawn from the data.

2.2.1 Duplicate Data

The very first thing we did was to check whether there were any duplicate observations, and there was not a single one.

2.2.2 Feature Engineering

Also, as shown in 2.1, there are predictors that are only available for Private individuals and predictors that are only available for Professional individuals. Thus, we need to make sure we have predictors available for both type of individuals every time.
There is a linear combination of *Amount of the monthly rent of the household*, *Monthly amount of the mortgage Amount of the monthly allowance* and *Monthly amount various household commitments* that compute *Amount of monthly income*:

$$mt\_rev = mt\_sal\_men + rev\_men\_autr + mt\_alloc\_men \cdot \frac{1 + \frac{1}{nb\_pers\_chg+1}}{2}$$

Hence we end up deleting all 4 predictors available only for Private individuals in order to get *Amount of monthly income* which is available for both type of individuals.

There is also a linear combination of *Amount of the monthly rent of the household*, *Monthly amount of the mortgage* and *Monthly amount various household commitments* that compute *Amount of charges*:

$$mt\_charges = mt\_loy\_men\_mena + mt\_men\_pre\_immo + mt\_men\_eng\_mena$$

Again we end up deleting all 3 predictors available only for Private individuals in order to get *Montant des charges* which is available for both type of individuals.

There is a last linear combination between *Montant de l'échéance* and *Prix du véhicule* that computes *Part de l'échéance*:

$$part\_loyer = \frac{mt\_ech}{mt\_ttc\_veh} \cdot 100$$

Finally, we end up deleting all 2 predictors available only for Private individuals in order to get *Part de l'échéance* which is available for both type of individuals.

Moreover, as *Filing indicator for PROs* and *Filing indicator for PRIs* bring the same kind of information but only for each type of individuals (Professional and Private respectively), we decided to create a dummy variable *fichage*, with modalities **0** if no indication or considered as "Moyen" and **1** if is it considered as "Mauvais" or "Très Mauvais". Hence, we end up with one predictor instead of two.
Finally, we created the predictor *support time* which is nothing but the time difference (in month) between the month of demand and the month of the date of entry into management.

2.2.3 Modalities regroupment

For some predictors, we have to deal with a lot of modalities. Some modalities were rare (occurence of the modality is low : $<1\%$) and we have to regroup them into an other modality like a modality named *Other* for example. The predictors that we have to regroup some modalities were: *CSP*, *secteur_* , *ETAT_ CIVIL* and *MODE_ HABI*.

The modifications made were:

- *CSP*: 16 modalities that we reduce to 5 modalities. Rare modalities were regrouped in a modality named *Other*.

- *secteur_*: 17 modalities that we reduce to 10 modalities. Rare modalities were regrouped in the existent modality *Other*.
- *ETAT_CIVIL*: 6 modalities that we reduce to 2 modalities. The modalities *Married* and *Common-law couple* were grouped together and the other modalities were groupe in a modality named *Single, widower....*
- *MODE_HABI*: 4 modalities that we reduce to 2 modalities. The modalities *Occupant*, *Hosted* and *Living accomodations* were grouped in a modality named *Non-owners*.

2.2.4 Missing Values

In case of absence of information on a predictor, we observe a missing value. Nevertheless, this predictor could be a carrier of information, removing the predictor would just make us lose all the information available. One way to deal with this problem is to create a missing value indicator class if the variable is sufficiently filled.

The particularity of our dataset is that it include a lot of missing values but only for some predictors. For example, *Cotation de la Banque de France* contains 25.555 missing values out of the 68.538 observations. Hence, we decided to make this predictor a dummy variable with the following modalities: **0** if we have no information (missing value) and **1** if we have.

2.3 Outliers

Outliers are extreme values that deviate from other observations on data , they may indicate a variability in a measurement, experimental errors or a novelty. In other words, an outlier is an observation that diverges from an overall pattern on a sample.
Outliers can introduce bias in the estimation of models. However, the discretization of the variables limit their impacts. Because of discretization, we will not take in consideration outliers in quantitative predictors. Indeed, outliers are important as removing them might skew our results.

Nevertheless, we decided to:

- remove observations with *customer age* having a value less than 18. Indeed, according to the law, a minor is not allowed to open a bank account alone, to invest money or to take credit.
- remove observations with *share of maturity* inferior to 0 as it is impossible.
- remove observations with a *percent contribution* superior to 100% as it is impossible.
- remove *evaluation of the quotation* as this predictor is subjective.
- remove the *evaluation of the payment behavior* predictor as we have *copot_* which basically tells the same information and is available for both type of individuals.
- remove *identifier of the encrypted contract* and *identifier of the encrypted client* for our model as this is just referring to IDs thus not important for modeling.
- remove *number of outstanding payments* and *number of outstanding payments from the last 12 months* as we only have those data for private individuals.

2.4 Summary

Table 2.1: Data Cleansing

Table 2.2: Before Cleaning

	Observations	Variables
Features Shape	68538	42
Target Shape	68538	1

Table 2.3: After Cleaning

	Observations	Variables
Features Shape	66510	26
Target Shape	66510	1

We end up with:

- a clean dataset to work with.
- 66510 observations.
- 26 predictors and 1 target variable.

Our clean dataset is about 97.04% of the raw dataset.

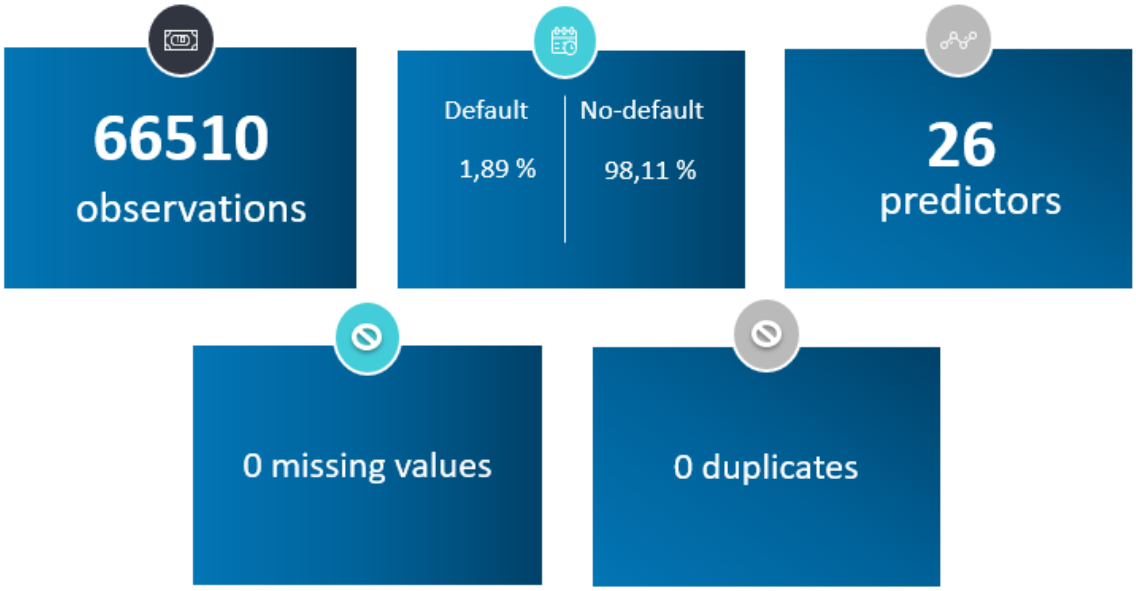

Figure 2.1: Dataset Summary

Chapter 3

Data Pre-processing

Data pre-processing is a data mining technique which is used to transform the raw data in a useful and efficient format. This latter involves transforming raw data into an understandable format. Real-world data is often incomplete, inconsistent, and/or lacking in certain behaviors or trends, and is likely to contain many errors. Data preprocessing is a proven method of resolving such issues.

3.1 Data Transformation

3.1.1 Feature Selection

To build a statistical model, it is first necessary to select the features which allow the best discrimination of our target variable, WE18. There are several ways to select features and we will use for quantitative predictors a Kruskall-Wallis test or a Student test and a Khi-2 test for qualitative predictors. Before that, we need to know what is the distribution of our quantitative predictors. Indeed, if the distribution is (supposed) to be normal, a Student test is sufficient. However if the distribution is not normal, a Kruskall-Wallis test is needed.

Here is the distribution of the quantitative predictors:

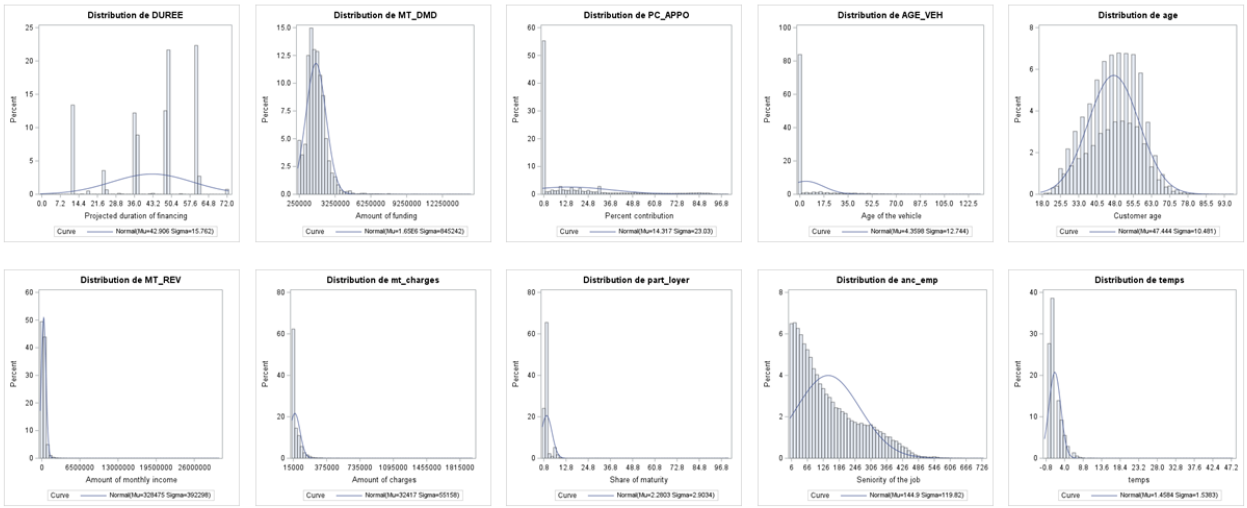

Figure 3.1: Evolution of the default rate per month

It is clear that unless for *age*, our quantitative predictors are not normally distributed. Hence a Kruskall-Wallis test is needed. Most of our quantitative predictors are skewed to the right due to outliers. However, discretization in a next step will (help) remove such effects.

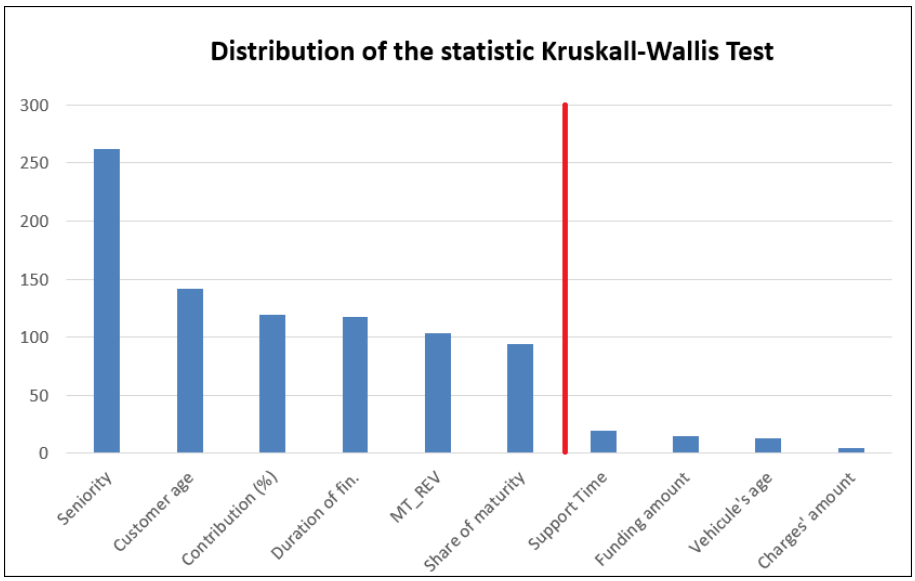

Figure 3.2: Kruskall-Wallis test

The red line indicates the threshold[1]at which we decide if or if not we keep the predictors. Predictors to the right of the red line are kept and those to the left are removed from our dataset. Hence, we are removing 4 predictors. Note that a high value of the khi-2 statistic shows that there is a strong connection between the variable and the default risk.

3.1.2 Discretization

In statistics and machine learning, **discretization** refers to the process of converting or partitioning continuous attributes, features or variables to discretized or nominal attributes/features/variables/intervals. Whenever continuous data is discretized, there is always some amount of discretization error. The goal is to reduce the amount to a level considered negligible for the modeling purposes at hand. This technique is useful in many cases as it allows us to limit the impact of outliers.

Achieving a good scoring model starts by ensuring stability over time in your model. Indeed, the more you discretize (i.e the more classes you have for each predictor) the better you will model risk. Nevertheless, in the industry you want to make sure the model is viable for quite some times so that you will not have to calibrate the model every day. Therefore, there is a **stability - accurary tradeoff** in the industry. Thus, we want to make sure we have stable predictors over time even tough we will probably lose some accurary.

However, when using the SAS Macro in order to discretize our quantitative variables, we saw that they were not stable over the time period. Hence, we had to modify and change the modalities in order to preserve stability in our model. One could also change from the statistical discretization if the intervals does not correspond to reality or does not mean anything (i.e one will prefer 48 months (4 years) than 42 months for interpretability sake). Indeed, sometimes it is better to model classes by yourself in order to gain in interpretation even tough those latter does not correspond to the best classes ever (statistically speaking).

Beneath this page, you can see the classes our SAS Macro got us by decile cutting (see Table 3.2, 3.5, 3.8, 3.11, 3.14 and 3.17) and the classes we chose (see Table 3.3, 3.6, 3.9, 3.12, 3.15 and 3.18) and their effects in terms of Risk & Volume stability. You can clearly see that the correction we made to those classes are **impactful** in term of stability. Indeed, there is no more cross line between different classes resulting in a much more **robust** model.

[1]The treshold has been arbitrarily chosen. However we can see a clear gap between Share of maturity and Support Time (the Khi-2 statistic goes from 93.63 to 19.24).

3.1.2.1 Homogeneity of Class individuals and Heterogeneity of Class Risk

Table 3.1: Projected duration of financing (**DUREE**)

Table 3.2: Macro

Class	Interval
1	$<$ 48 months
2	48 months $\leq$ **2** $<$ 60 months
3	$\geq$ 60 months

Table 3.3: Corrected

Class	Interval
1	$<$ 60 months
2	$\geq$ 60 months

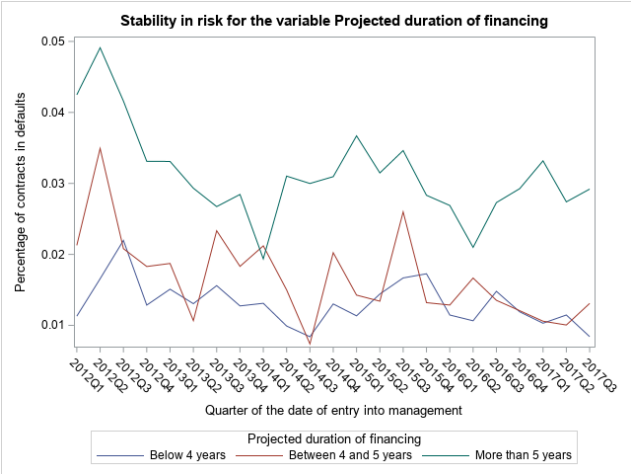

Figure 3.3: Risk stability - Macro discretization

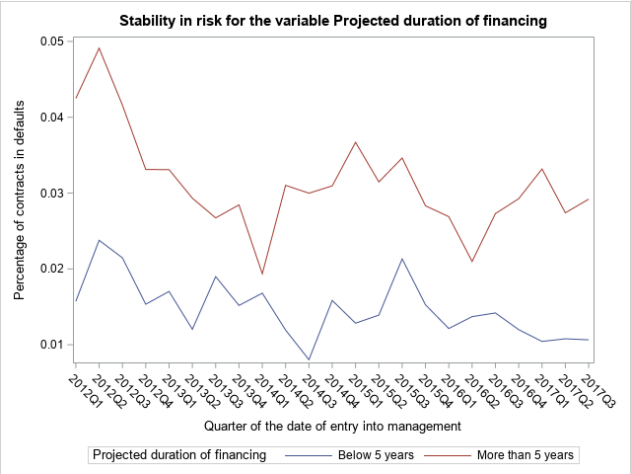

Figure 3.4: Risk stability - Corrected discretization

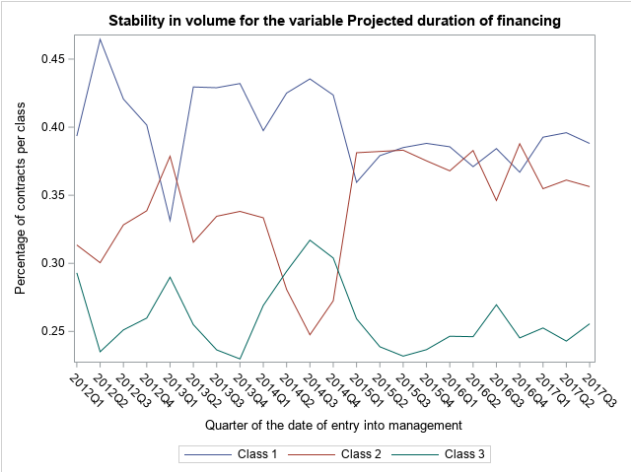

Figure 3.5: Volume stability - Macro discretization

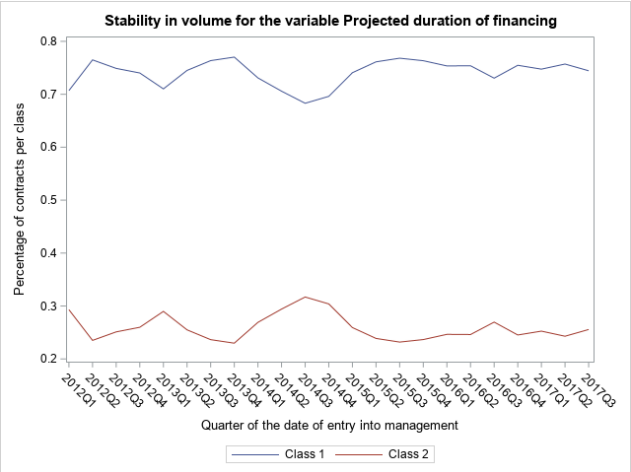

Figure 3.6: Volume stability - Corrected discretization

Table 3.4: Âge (**age**)

Table 3.5: Macro

Class	Interval
1	$<$ 43 years
2	43 years $\leq$ **2** $<$ 54 years
3	$\geq$ 54 years

Table 3.6: Corrected

Class	Interval
1	$\leq$ 43 years
2	43 years $<$ **2** $<$ 58 years
3	$\geq$ 58 years

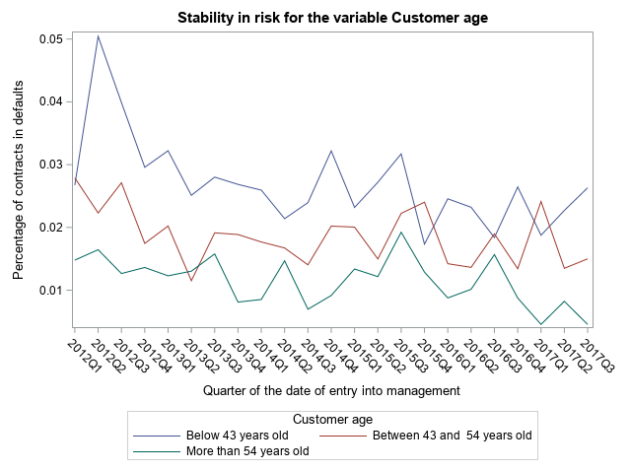

Figure 3.7: Risk stability - Macro discretization

Figure 3.8: Risk stability - Corrected discretization

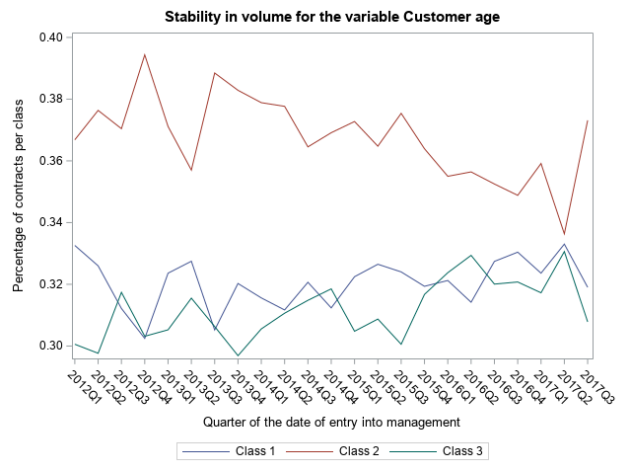

Figure 3.9: Volume stability - Macro discretization

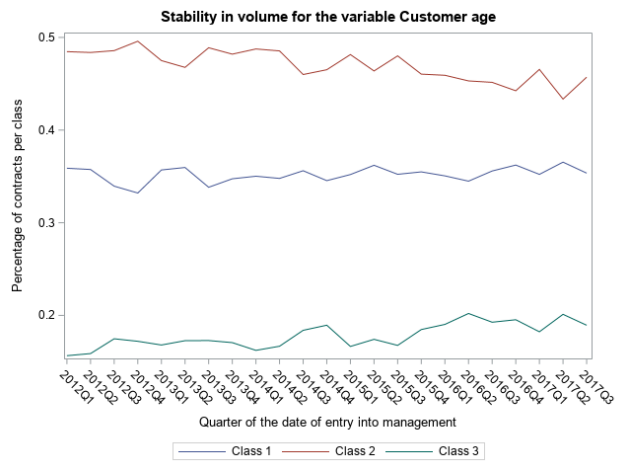

Figure 3.10: Volume stability - Corrected discretization

Table 3.7: Percent contribution (**PC_APPO**)

Table 3.8: Macro

Class	Interval
1	$< 20\ \%$
2	$20\ \% \leq \mathbf{2} < 35\ \%$
3	$\geq 35\ \%$

Table 3.9: Corrected

Class	Interval
1	$< 20\ \%$
2	$20\ \% \leq \mathbf{2} < 35\ \%$
3	$\geq 35\ \%$

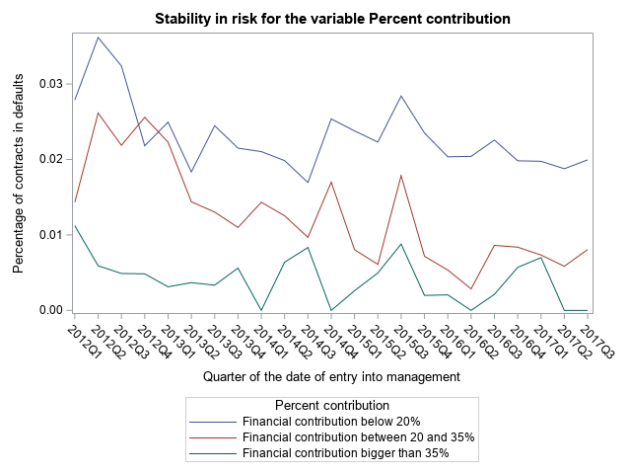

Figure 3.11: Risk stability - Macro discretization

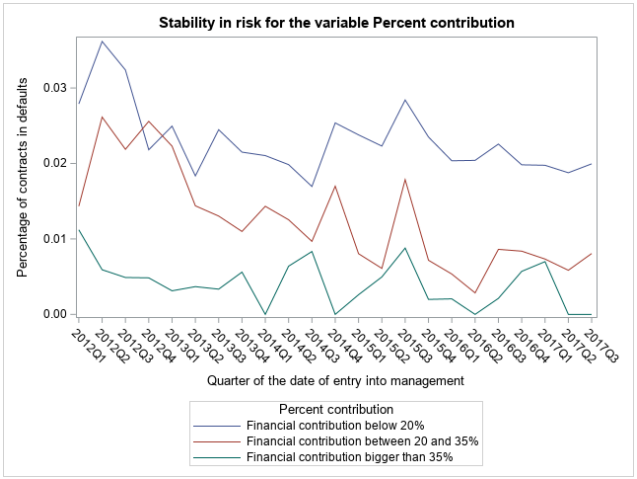

Figure 3.12: Risk stability - Corrected discretization

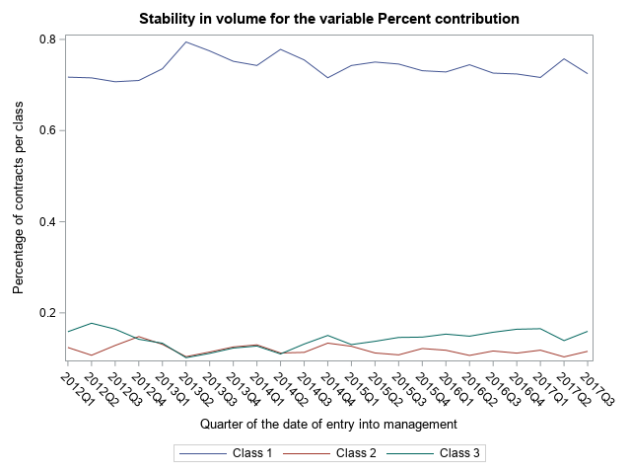

Figure 3.13: Volume stability - Macro discretization

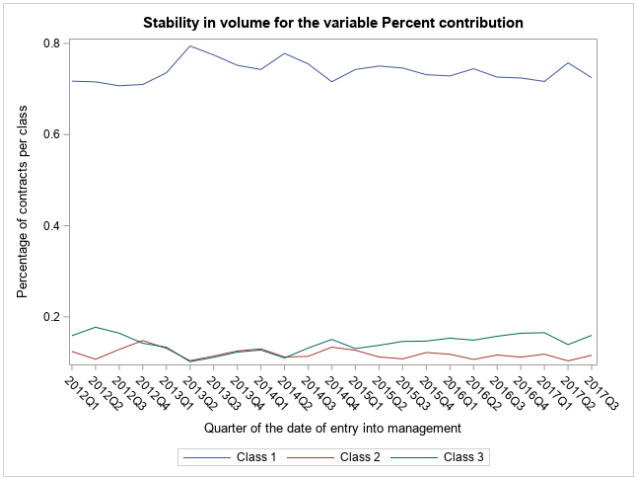

Figure 3.14: Volume stability - Corrected discretization

Table 3.10: Amount of monthly income (**MT_REV**)

Table 3.11: Macro

Class	Interval
1	< 2.500 €
2	2.500 €$\leq$ **2** $< 6.138{,}67$ €
3	$\geq 6.138{,}67$ €

Table 3.12: Corrected

Class	Interval
1	< 3.000 €
2	≥ 3000 €

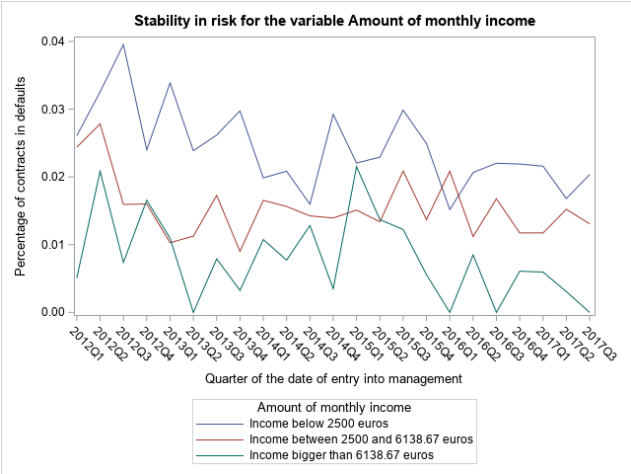

Figure 3.15: Risk stability - Macro discretization

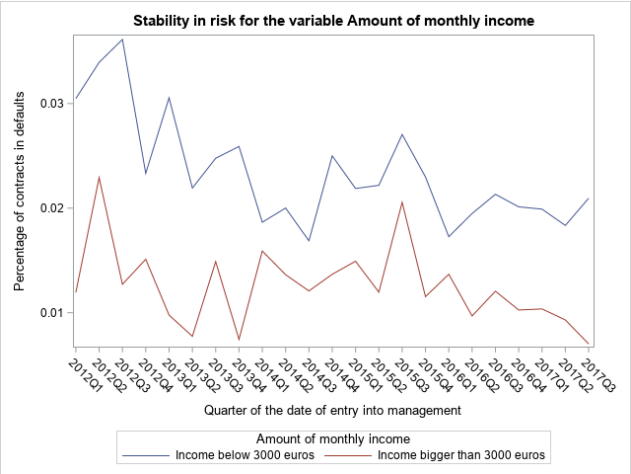

Figure 3.16: Risk stability - Corrected discretization

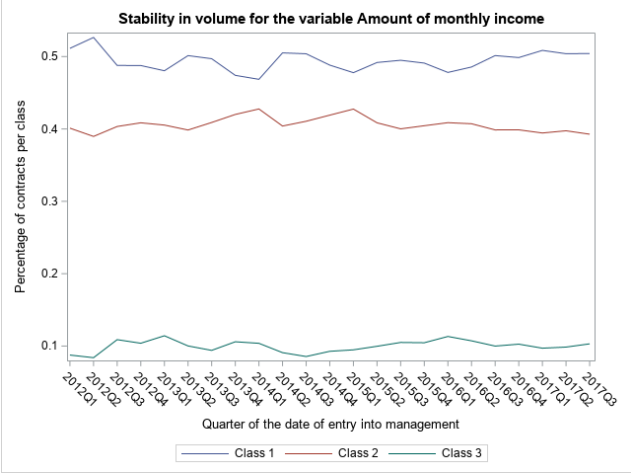

Figure 3.17: Volume stability - Macro discretization

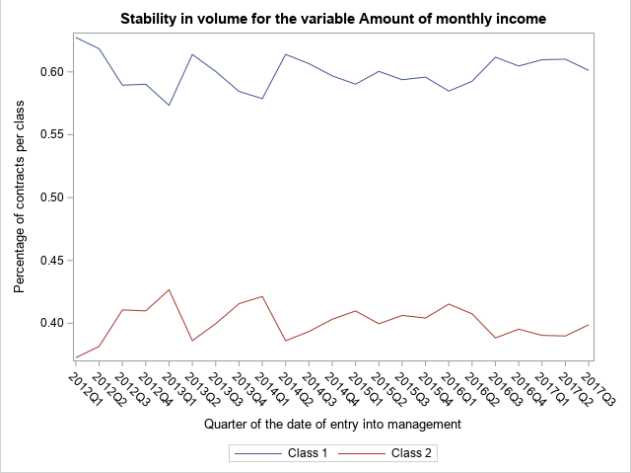

Figure 3.18: Volume stability - Corrected discretization

Table 3.13: Share of maturity (**part_loyer**)

Table 3.14: Macro

Class	Interval
1	< 1.21 %
2	1.21 % $\leq$ **2** < 1.78 %
3	≥ 1.78 %

Table 3.15: Corrected

Class	Interval
1	≤ 1.20 %
2	1.20 % $<$ **2** < 1.80 %
3	≥ 1.80 %

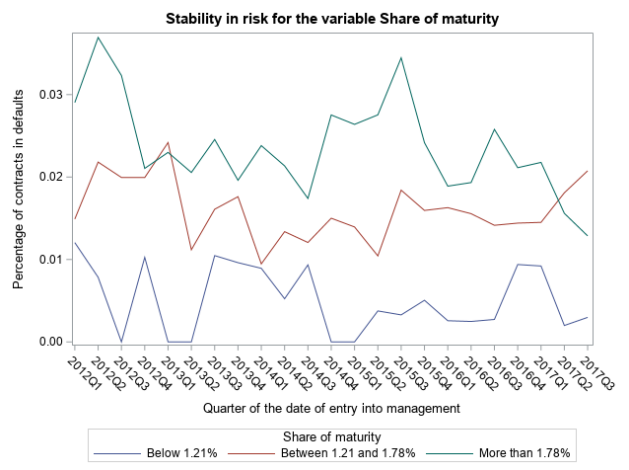

Figure 3.19: Risk stability - Macro discretization

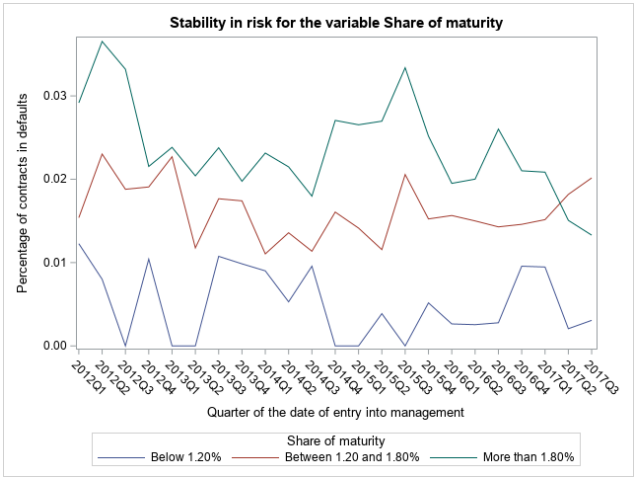

Figure 3.20: Risk stability - Corrected discretization

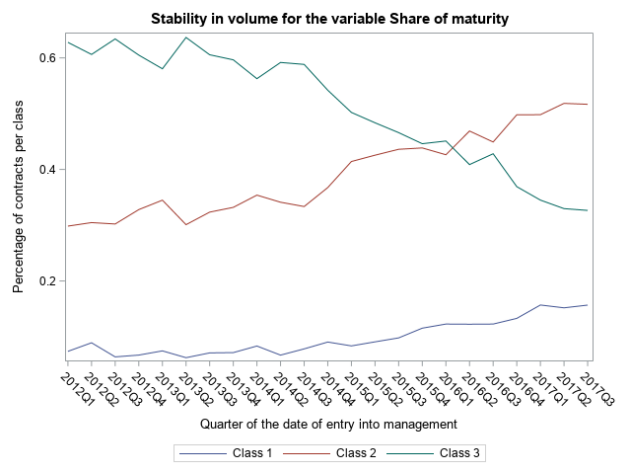

Figure 3.21: Volume stability - Macro discretization

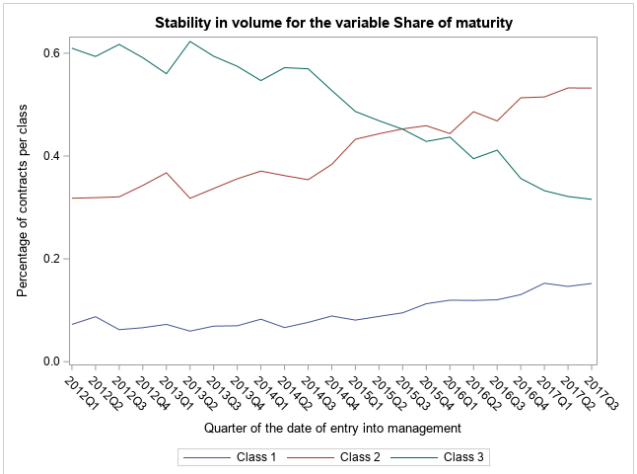

Figure 3.22: Volume stability - Corrected discretization

Table 3.16: Seniority of the job (**anc_emp**)

Table 3.17: Macro

Class	Interval
1	$<$ 37 months
2	37 months $\leq$ **2** $<$ 146 months
3	$\geq$ 146 months

Table 3.18: Corrected

Class	Interval
1	$<$ 48 months
2	48 months $\leq$ **2** $<$ 144 months
3	$\geq$ 144 months

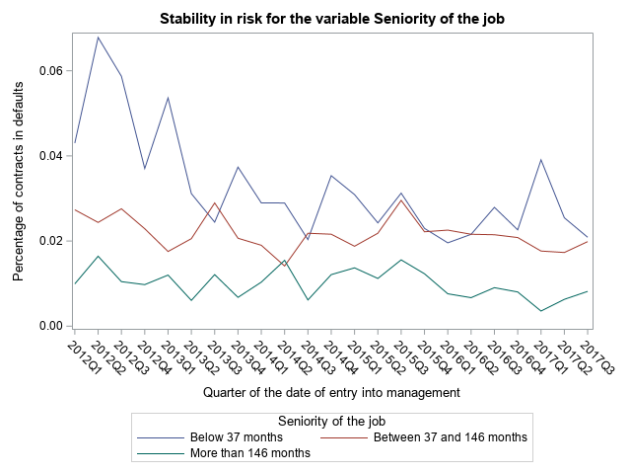

Figure 3.23: Risk stability - Macro discretization

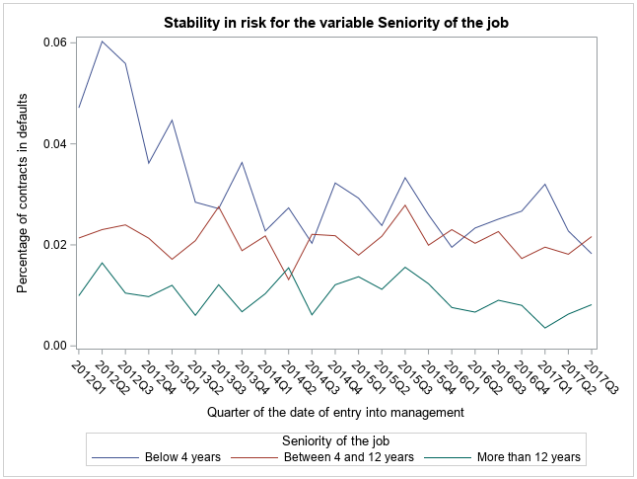

Figure 3.24: Risk stability - Corrected discretization

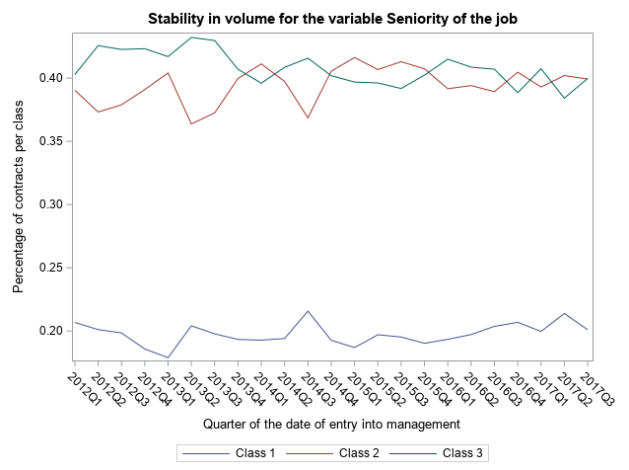

Figure 3.25: Volume stability - Macro discretization

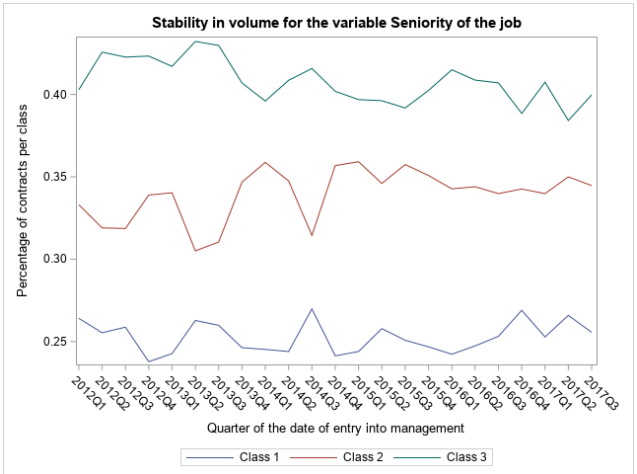

Figure 3.26: Volume stability - Corrected discretization

3.1.2.2 Cramér's V

In statistics, **Cramér's V** (sometimes referred to as Cramér's phi) is a measure of association between two nominal variables, giving a value between 0 and +1 (inclusive). It is based on Pearson's chi-squared statistic and was published by Harald Cramér in 1946. Cramér's V varies from 0 (corresponding to no association between the variables) to 1 (complete association) and can reach 1 only when each variable is completely determined by the other. This latter is defined as:

$$\phi_c = \sqrt{\frac{\chi^2}{N(k-1)}}$$

With ϕ_c Cramér's V, χ^2 the Pearson chi-square statistic, N the sample size involved in the test and k the lesser number of categories of either variable.

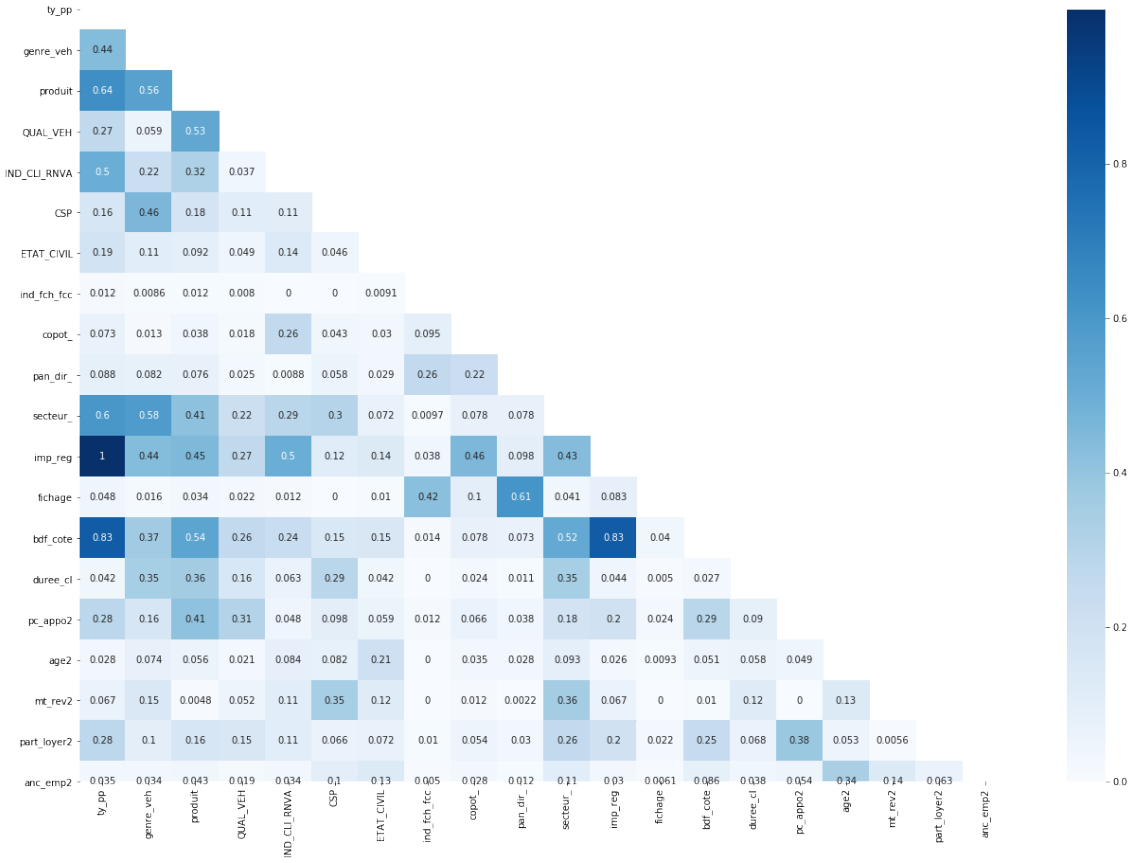

Figure 3.27: Cramér's V Matrix Before

We arbitrary decide to delete predictors that have a Cramér's V superior to 0.65. We can clearly see that the following predictors *imp_reg*, *produit* and *bdf_cote* are highly correlated with *ty_pp* (with a coefficient of 1, 0.84 and 0.83 respectively). Thus we decide to delete *ty_pp*. We can also see that *bdf_cote* is highly correlated with *imp_reg*. Hence we proceed to delete *imp_reg* as this is a predictor that we created earlier.

An alternative association measure for two nominal variables is the contingency coefficient. However, it's better avoided since its maximum value depends on the dimensions of the contingency table involved. For two metric variables, a Pearson correlation is the preferred measure.

3.2 Data Reduction

3.2.1 SMOTE

When the class distribution of the target variable is not uniform among the classes we say that we are dealing with **imbalanced** datasets. In our case, the proportion of individuals in default is 1.89%. Machine learning algorithms have trouble learning when one class dominates the other that is why we need to introduce a way for our algorithm to help him discriminate classes. In order to do so, we will use The Synthetic Minority Oversampling Technique aka **SMOTE**. In order to over samples rare events, SMOTE create additional synthetically observations of that event by using bootstrapping and k-nearest neighbor. The white paper can be found on arXiv. Because a picture means a thousand words, here is an example of Bordeline-SMOTE used in Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning.

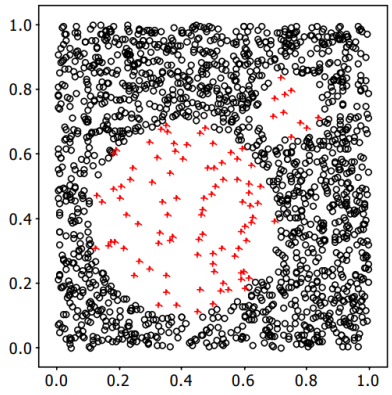

Figure 3.28: The original distribution of Circle data set

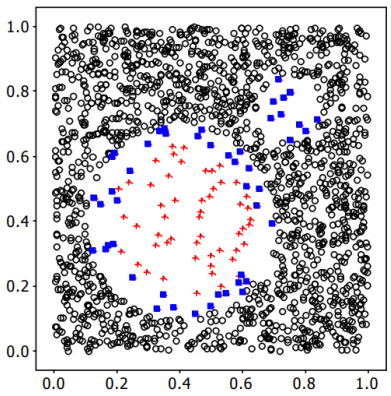

Figure 3.29: The borderline minority examples

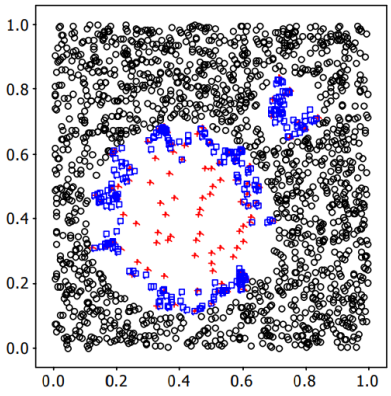

Figure 3.30: The borderline synthetic minority examples

Please note that we are introducing bias to our model as we are creating "fake" default individuals, but technically this is a great bias in a sense that we are helping our algorithm discriminate classes. In other words, we created correlation between our individuals but this correlation help us discriminate classes.

3.3 Summary

	Initial Sample	Subsample	SMOTE
Number of Observations	66.510	11.340	51.660
Number of Default	1.260	1.260	11.340
Default Ratio	1.89%	11.11%	21.95%

Chapter 4

Data Visualization

Data visualization is the graphical representation of information and data. By using visual elements like charts, graphs, and maps, data visualization tools provide an accessible way to see and understand trends, outliers, and patterns in data.

4.1 Default Risk over time

4.1.1 Histogram of Default Rate and Amount of Loans over the Period

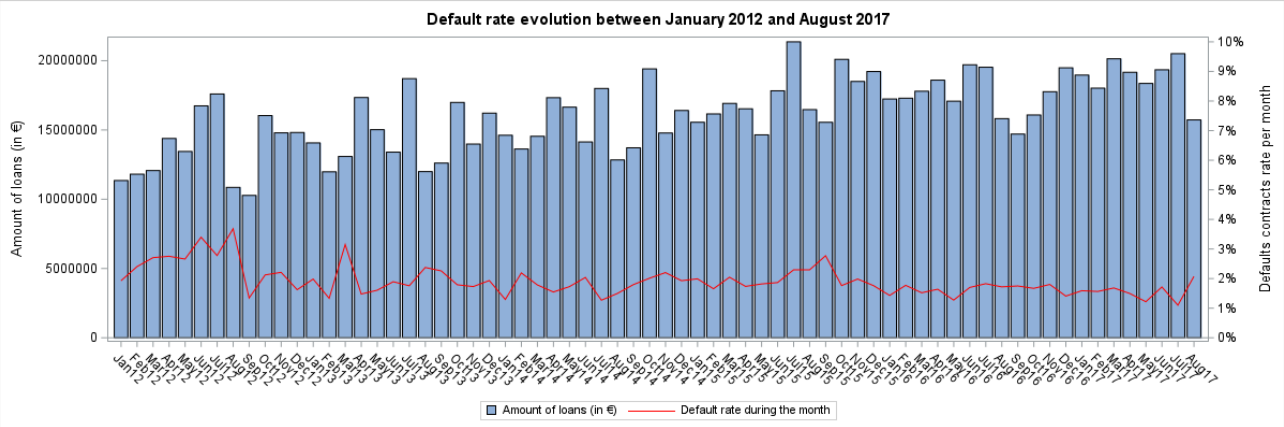

Figure 4.1: Evolution of the Default Rate & Amount of Loans per Month

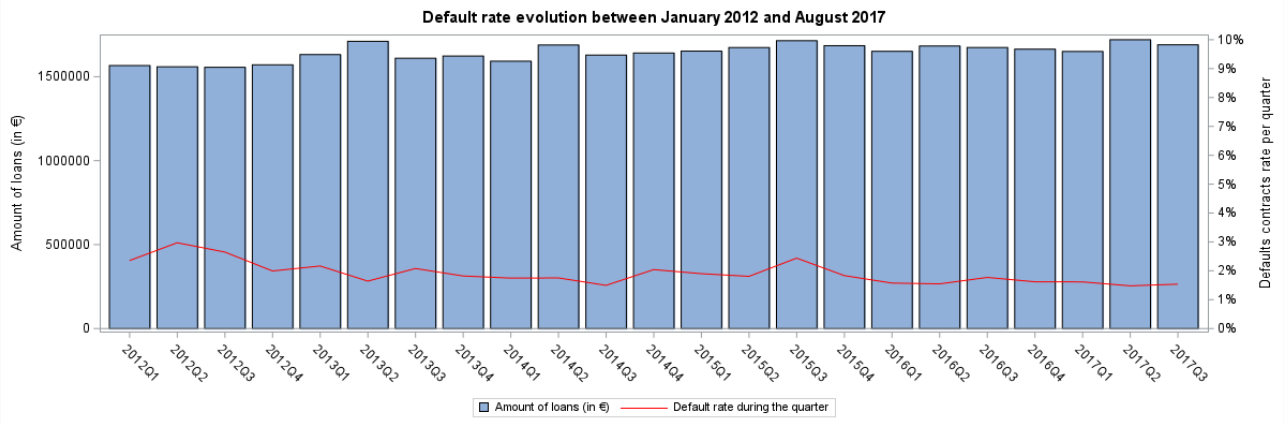

Figure 4.2: Evolution of Default Rate & Amount of Loans per Quarter

System stability or population stability compares the data sample the model was developed on with a more recent data sample on which the model has been used. The discriminatory power of a model is based on information contained in the development dataset, and hence large variances from this may cause model performance to deteriorate, or make the model unfit for purpose.

According to the two histograms above, we can clearly see a smoothing of the default rate curve as well as a reconciliation of loan amounts around 1.500.000 € when you take in consideration a quarter-period instead of a month-period. Those plots ensures us **stability** of our target variable over the period which is really important if you want to model the default risk.

4.2 Variable Grouping

4.2.1 Socio-Professional Class

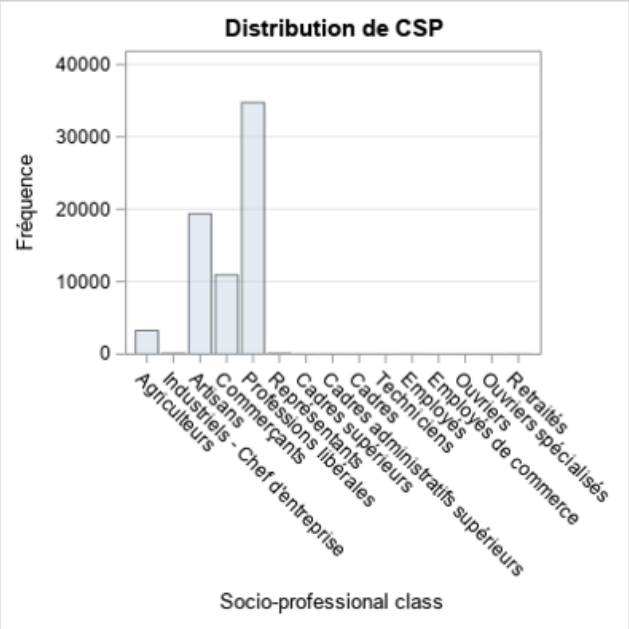

Figure 4.3: Socio-Professional Class - Before

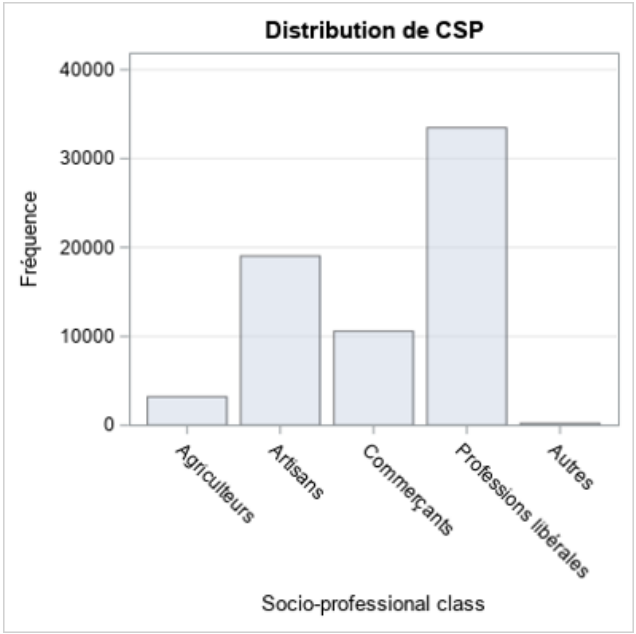

Figure 4.4: Socio-Professional Class - After

4.2.2 Activity Area

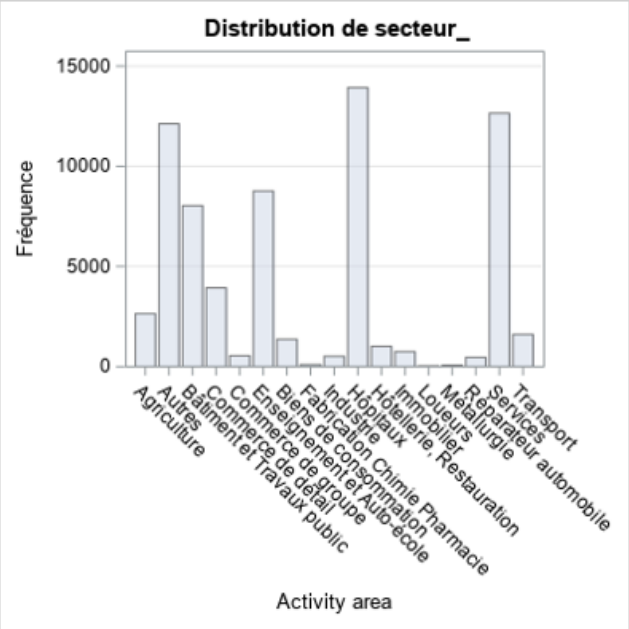

Figure 4.5: Activity Area - Before

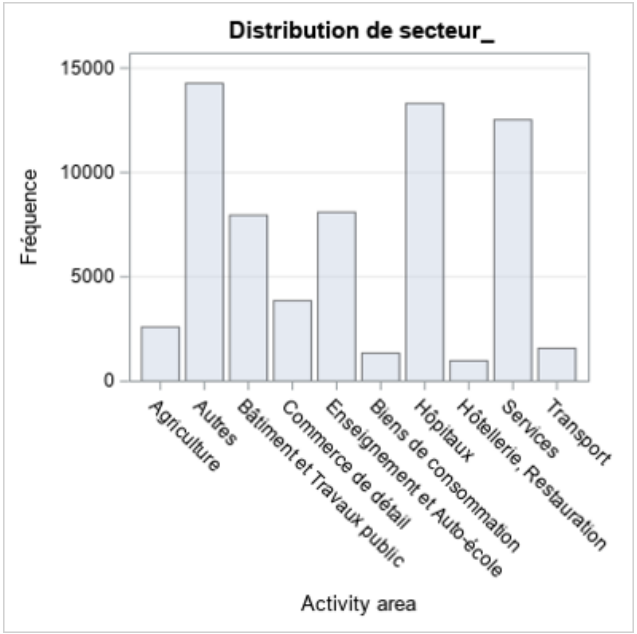

Figure 4.6: Activity Area - After

4.2.3 Civil Status

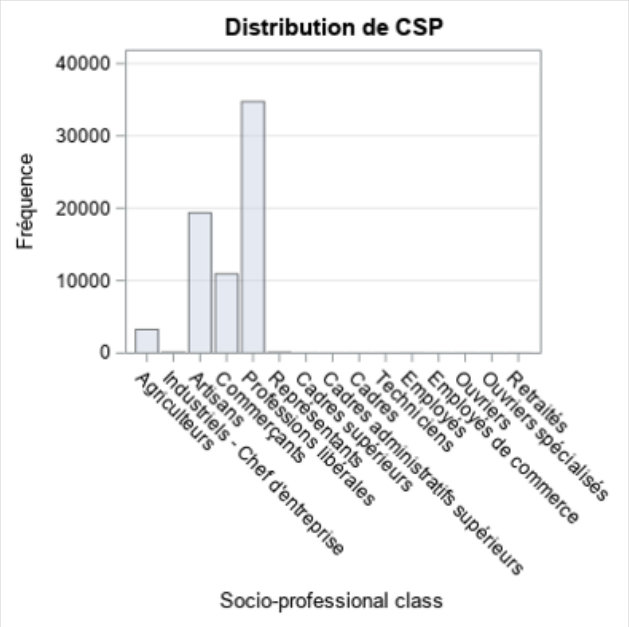

Figure 4.7: Civil Status - Before

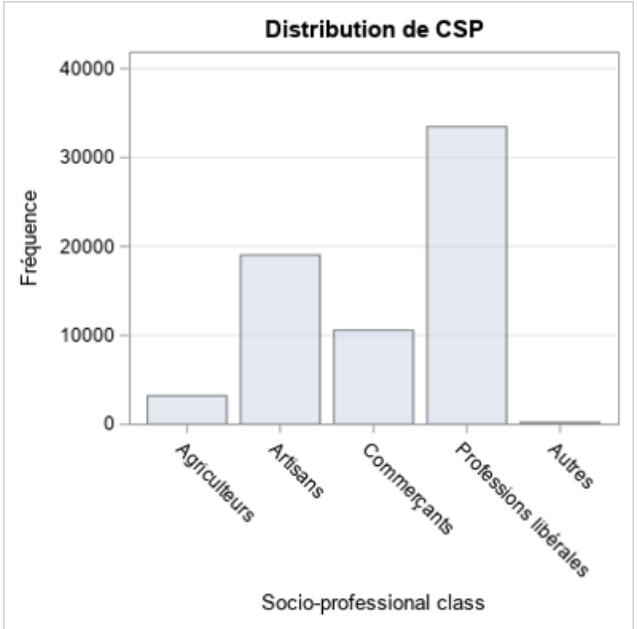

Figure 4.8: Civil Status - After

4.2.4 Living Mode

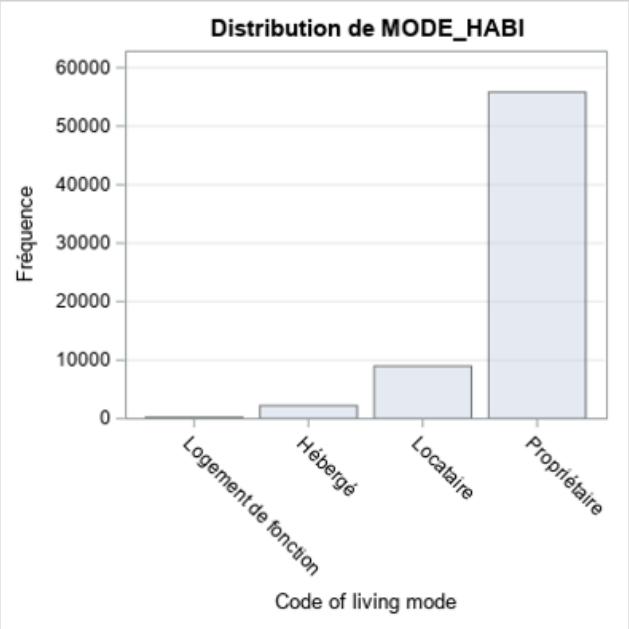

Figure 4.9: Living Mode - Before

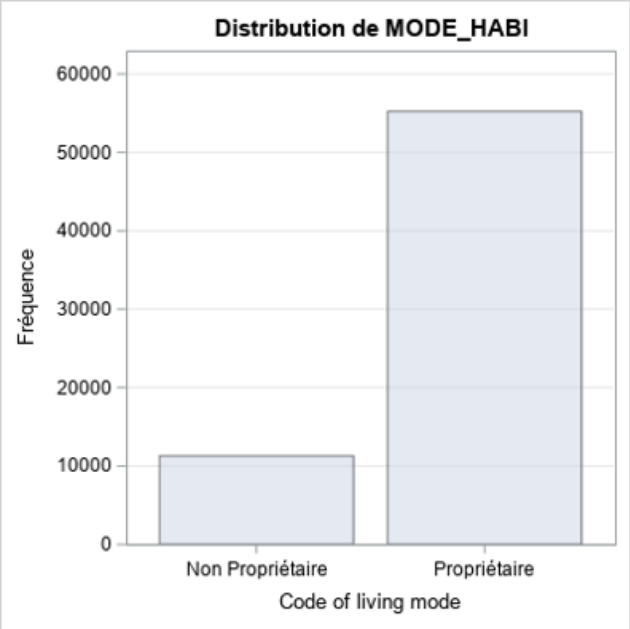

Figure 4.10: Living Mode - After

Chapter 5

Methodology

5.1 Preparation

5.1.1 One-Hot-Encoding

One-hot-encoding is a common technique used to work with categorical variables.
Indeed we had to create dummies for those latter. `pandas` as a useful and easy implementation of it.

5.1.2 Training Set & Test Set

We decided to split the dataset as follow: 75% for the Training Set and 25% for the Test Set.

	Observations	Variables
Training Features Shape	50064	50
Training Target Shape	50064	1
Testing Features Shape	16688	50
Testing Target Shape	16688	1

Figure 5.1: Training and Test - RAW Dataset

	Observations	Variables
Training Features Shape	8505	55
Training Target Shape	8505	1
Testing Features Shape	2835	55
Testing Target Shape	2835	1

Figure 5.2: Training and Test - Subsample Dataset

	Observations	Variables
Training Features Shape	38745	55
Training Target Shape	38745	1
Testing Features Shape	12915	55
Testing Target Shape	12915	1

Figure 5.3: Training and Test - SMOTE Dataset

5.2 Random Forest

Random Forest is an ensemble Machine Learning technique capable of performing both regression and classification tasks using multiple decision trees and a statistical technique called bagging. Bagging along with boosting are two of the most popular ensemble techniques which aim to tackle high variance and high bias. The first algorithm for random decision forests was created by Tin Kam Ho. An extension of the algorithm was developed by Leo Breiman.
Random for two reasons:

- Random sampling of training observations when building trees.
- Random subsets of features for splitting nodes.

5.2.1 Random sampling of training observations

The samples are drawn with replacement, known as bootstrapping, which means that some samples will be used multiple times in a single tree.
For our analysis we will use the `Sklearn` library. In `Sklearn`, implementation of Random forest the subsample size of each tree is always the same as the original input sample size but the samples are drawn with replacement if `bootstrap = True`. If `bootstrap = False` each tree will use exactly the same dataset without any randomness.

5.2.2 Random subsets of features for splitting nodes

Random Forest with random features is formed by selecting at random, at each node, a small group of input variables to split on. In `Sklearn` this can be set by specifying `max features = sqrt(n_features)`.
If `max features = auto` then the algorithm will choose by itself the best max number of features.

5.2.3 Other Hyperparameters

There are several other hyperparameters for the Random Forest. Here are the ones we will focus on:

- `max_depth` = The maximum depth of the tree. If None, then nodes are expanded until all leaves are pure or until all leaves contain less than min samples split samples.
- `min_samples_leaf` = The minimum number of samples required to be at a leaf node. A split point at any depth will only be considered if it leaves at least `min_samples_leaf` training samples in each of the left and right branches. This may have the effect of smoothing the model, especially in regression.
- `min_samples_split` = The minimum number of samples required to split an internal node.
- `n_estimators` = The number of trees in the forest.

5.2.4 Hyperparameter Estimation

Grid-search is used to find the optimal hyperparameters of a model which results in the most 'accurate' predictions. Grid search is an approach to hyperparameter tuning that will methodically build and evaluate a model for each combination of algorithm parameters specified in a grid.
We used **Randomized Search** which is implemented in `Sklearn` to find optimal hyperparameters as discussed in 4.2.1, 4.2.2 & 4.2.3.
The Randomized Search and the Grid Search explore exactly the same space of parameters. The result in parameter settings is quite similar, while the run time for randomized search is drastically lower. The performance is slightly worse for the randomized search, though this is most likely a noise effect and would not carry over to a held-out test set.

Here is the Grid we tried with Optimal Value being the value chosen by the Randomized:

Hyperparameter	Range	Optimal Value
Bootstrap	[True, False]	True
Deepness	[10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, None]	None
Max. Features	[Auto, Sqrt]	Auto
Min. Sample Leaf	[1, 2, 4]	1
Min. Sample Split	[2, 5, 10]	2
Number of estimators	[200, 400, 600, 800, 1000, 1200, 1400, 1600, 1800, 2000]	400

Figure 5.4: Grid for the Random Forest Randomized Search

We fitted 3 folds (`cv = 3`) for each of 100 candidates (`n_iter = 100`), resulting in 300 fits.
Randomized Search took 5.9 minutes to execute whereas Grid Seach did not even finish after 1.25 hours. We observed an increase of about 7% on average for both the Gini index and 10/X index when using the parameters chose by the Randomized Search.

5.3 Support Vector Machine

Support Vector Machine[1] or **SVM** are a set of supervized learning methods that are used for classification analysis. It has been developed in the 1990's from Vladimir Vapnik's theorical considerations. It has been quickly adopted for its capacity to treat large amount of data, the low number of parameters it requires and the fiability of the results it provides.

Basically, the idea is to find a decision boundary to split data in regions corresponding to the classes of our objective variable. The classifier attributes a class to our observation according to the region it is in. In linear separable cases, SVM consists in finding the separating hyperplan that classify correcly and that maximize the margin. There is obviously an infinity numbers of separating hyperplans. In order to choose the best hyperplane, we select the one that maximizes the distance from it to the nearest data point on each side. This optimal decision boundary is called maximum-margin hyperplane, the margin being the distance between the hyperplane and the nearest data points, known as the support vectors.

5.3.1 Kernels

In reality, datasets are almost never linearly separable. That is why it is impossible to obtain a linear frontier that will 100% correctly separate each observations according to the class it belongs to. If that is the case, we have to use soft margin, authorizing some observations to be misclassified or to be inside the margin. Most of problems fall under non-linear separation. The Kernel trick rely on the fact that: "A complex pattern-classication problem, cast in a high-dimensional space nonlinearly, is more likely to be linearly separable than in a low-dimensional space, provided that the space is not densely populated." (T.M. Cover, 1965)

In order to improve the space of data, we resort to a kernel function that will create new axis by combining existing ones. The kernel function has to respect Mercers theorem's conditions and does not require an explicit knowledge of the transformations used to transform the space of data.

In the `scikit-learn` library, you can instantiate 4 different kernels:

- Linear Kernel, $K(X, Y) = X^T Y$
- Polynomial Kernel, $K(X, Y) = (\lambda \cdot X^T Y + r)^d, > 0$
- Radial basis function (RBF) Kernel, $K(X, Y) = \exp(\|X - Y\|^2 / 2\sigma^2)$ which in simple form can be written as $\exp(-\lambda \cdot \|X - Y\|^2), > 0$
- Sigmoid Kernel; $K(X, Y) = \tanh(\lambda \cdot X^T Y + r)$ which is similar to the sigmoid function in logistic regression

[0]Here are two Shiny Applications (1 and 2) in order to understand SVMs and their application to a Credit Fraud dataset.

5.4 Metrics of comparison

5.4.1 Gini index

The **Gini index**, developed by the Italian statistician and sociologist Corrado Gini in 1912, is a measure of statistical dispersion and is the most commonly used measurement of inequality. It is most often used in economics to measure how far a country's wealth or income distribution deviates from a totally equal distribution. The Gini index derive directly from the AUC. Hence we can compute it as follow:

$$Gini = 2 \cdot \texttt{AUC} - 1$$

The coefficient ranges from 0 (or 0%) to 1 (or 100%), with 0 representing perfect equality and 1 representing perfect inequality.

5.4.2 10/X index

The **index 10/X** represents the proportion of default captured if we take only the 10% of individuals with the lowest score. The higher the efficiency of the score, the higher the proportion.
Please note that this index depends on how much your dataset is imbalanced. Indeed, if you only have 0.01% of default, then your 10/X index will tend to 0. On the contrary, if your dataset has 99.9% of default, then your 10/X index will tend to 1. In other words, the more default you have in your dataset, the higher the 10/X index will be. This is what we end up showing in our Chapter 6.

In order to compute the 10/X index for our Machine Learning models, we decided to proceed as follow[0]:

1. Fit the model on the training dataset.
2. Make the predictions on the test dataset.
3. Get the prediction probability of default.
4. Take the 10% with the highest probability in an array A (this is equivalent to take the 10% with the lowest Score for the Logistic Regression).
5. Merge this array with the corresponding true identifier in the Test Set (ie. if he is really in default or not).
6. Compute the 10/X index by taking the frequency of default in the 10% with the highest probability.

[0]The code can be found on the GitHub Repository (cf. Annexes)

Chapter 6

Results

6.1 Scoring Grid - Logistic Regression

With the logistic regression, we were able to make a scoring grid for our three samples. For our study, we split our data in a training set that contains 75% of our samples and the rest of our observations were used for our testing. We used our training sample to build our scoring grid.

6.1.1 How to make a Scoring Grid

In order to create a Scoring Grid for our three datasets, we have to make a logistic regression. The logistic regression is necessary because we need the predictor's coefficient for every modalities that are present in our Scoring Grid.

Now that we have the predictor's coefficient, we have to calculate a number called $Delta(i, j)$:

$$Delta(i, j) = C(j, i) - min(j)$$

where $C(j, i)$ is the i-th modality of the j-th variable and $min(j)$ is the smallest predictor's coefficient of the j-th modality.

After that, we are able to compute a Score for each modality of a variable. When the Score of a modality is low, it means that this modality can easily discriminate contracts.

The Score that we obtain is between 0 and 100 and is computed as follow:

$$Score(i, j) = 100 * \frac{Delta(j,i)}{Delta(totality)}$$

where $Delta(totality)$ is the sum of the highest predictor's coefficient for each variable.

6.1.2 Scoring Grid - RAW dataset

Variables	Modalities	Percentages of defaults	Repartition of the population	Ponderation
Seniority of the job	Under 48 months	3%	25,1%	0.0000
	Between 48 and 144 months	2,1%	33,8%	3.0475
	Over 144 months	1%	41,1%	6.6467
Rating of the Banque de France	No Information	1,5%	37,8%	1.8436
	Information	2,1%	62,2%	0.0000
Payment behavior	Good	1,5%	92,3%	15.7038
	Medium	4,4%	1,3%	8.7393
	Bad	12%	3,4%	0.0000
Socio-professional class	Farmers	1,4%	4,7%	4.7912
	Craftmens	2,7%	28,8%	4.7336
	Others	3,4%	0,3%	0.0000
	Shopkeepers	3,4%	15,7%	4.2467
	Liberal professions	1%	50,5%	6.7343
Projected duration of financing	Under 60 months	1,5%	74,5%	5.0885
	Over 60 months	2,9%	25,5%	0.0000
Civil status code	Married	1,5%	50,9%	3.3434
	Single, widower...	2,3%	49,1%	0.0000
Filing indicator	No	1,9%	99,7%	6.2738
	Yes	11,1%	0,3%	0.0000
Renewing customer indicator	No	2%	80%	0.0000
	Yes	1,6%	20%	2.3862
FCC record indicator	No	1,9%	99,9%	14.6857
	Yes	14,3%	0,1%	0.0000
Code of living mode	Non-owners	4,1%	16,7%	0.0000
	Owners	1,4%	83,3%	5.6635
Share of maturity	Under 1,2%	0,3%	9,7%	7.7296
	Between 1,2 and 1,8%	1,8%	41,7%	2.4040
	Over 1,8%	2,3%	48,6%	0.0000
Percent contribution	Under 20%	2,4%	74%	0.0000
	Between 20 and 35%	1%	11,8%	2.9878
	Over 35%	0,4%	14,2%	4.1643
Product type	Traditional credits	2,2%	30%	2.5657
	Leasing contracts	1,6%	31,8%	4.6549
	Long-term rental	1,9%	38,2%	0.0000
Quality of the vehicle	Second hand vehicle	1,8%	83,1%	0.0000
	New vehicle	2,2%	16,9%	1.6482
Activity area	Farming	1,7%	3,9%	4.1396
	Others	1,8%	21,1%	4.6549
	Consumer goods	3,4%	1,9%	2.6232
	Construction industry	2,7%	12,2%	1.8895
	Retail business	4,1%	6%	4.4273
	Teaching and driving school	2%	12%	7.7619
	Hospitals	0,4%	20,3%	13.4336
	Hotels-Restaurants	6,5%	1,4%	1.8726
	Utilities	1,6%	18,8%	6.1489
	Transport	2,8%	2,4%	0.0000

6.1.3 Scoring Grid - Subsample dataset

Variables	Modalities	Percentages of defaults	Repartition of the population	Ponderation
Age	Under 43 years old	15,35%	34,1%	0.0000
	Between 43 and 58 years old	4,62%	47,8%	0.2317
	Over 58 years	0,99%	18,1%	2.6997
Seniority of the job	Under 48 months	16,2%	24,1%	0.0000
	Between 48 and 144 months	12,7%	35,5%	2.6626
	Over 144 months	6,3%	40,4%	7.8123
Payment behavior	Good	9,5%	91,9%	20.4663
	Medium	19,8%	6,6%	15.1750
	Bad	44,8%	1,5%	0.0000
Socio-professional class	Farmers	4,8%	4,7%	1.7569
	Craftmens	16,5%	29,5%	1.3557
	Others	8.6%	0,5%	11.4206
	Shopkeepers	15,9%	15,5%	0.0000
	Liberal professions	6,6%	49,8%	5.3431
Projected duration of financing	Under 60 months	9%	73,3%	3.4588
	Over 60 months	16,4%	26,7%	0.0000
Civil status code	Married	8,6%	51,6%	4.2078
	Single, widower...	13,7%	48,4%	0.0000
Filing indicator	No	11%	99,6%	11.6124
	Yes	35,7%	0,4%	0.0000
FCC record indicator	No	11%	99,8%	8.0495
	Yes	33,3%	0,2%	0.0000
Code of living mode	Non-owners	24,2%	15,4%	0.0000
	Owners	6,7%	84,6%	5.8177
Share of maturity	Under 1,2%	3,3%	10,4%	3.8599
	Between 1,2 and 1,8%	9,6%	40,7%	2.1511
	Over 1,8%	13,8%	48,9%	0.0000
Percent contribution	Under 20%	13,3%	74,5%	0.0000
	Between 20 and 35%	6,8%	11%	4.2630
	Over 35%	1,6%	14,5%	7.8866
Activity area	Farming	4,5%	4,2%	5.2073
	Others	10,2%	21,4%	3.7476
	Consumer goods	19,8%	2,4%	5.5964
	Construction industry	17,6%	11,7%	0.8961
	Retail business	14,7%	6,2%	4.7401
	Teaching and driving school	11,4%	11,7%	4.6261
	Hospitals	4,3%	19,4%	12.7085
	Hotels-Restaurants	26,2%	1,2%	1.3331
	Utilities	9,8%	19,1%	5.5338
	Transport	16,3%	2,7%	0.0000

6.1.4 Scoring Grid - SMOTE dataset

Variables	Modalities	Percentages of defaults	Repartition of the population	Ponderation
Age	Under 43 years old	25,7%	36,3%	0.0895
	Between 43 and 58 years old	23,1%	46,2%	0.0000
	Over 58 years	9%	17,5%	3.8891
Seniority of the job	Under 48 months	29,5%	25,6%	0.0000
	Between 48 and 144 months	27,8%	33,6%	1.3397
	Over 144 months	9,9%	40,8%	6.3624
Rating of the Banque de France	No Information	15,7%	38,7%	2.4919
	Information	25,4%	61,3%	0.0000
Payment behavior	Good	14,1%	92,4%	15.3909
	Medium	56,3%	6,2%	7.5015
	Bad	78,1%	1,4%	0.0000
Socio-professional class	Farmers	17%	5,2%	1.4150
	Craftmens	32,2%	27,9%	1.4758
	Others	8,6%	0,3%	6.9074
	Shopkeepers	30,3%	15,8%	0.0000
	Liberal professions	12%	50,8%	3.8502
Projected duration of financing	Under 60 months	18%	74,2%	1.1866
	Over 60 months	31,4%	25,8%	0.0000
Filing indicator	No	21,1%	99,6%	3.9739
	Yes	80,1%	0,4%	0.0000
Type of vehicle	Particular vehicle	18,5%	82,2%	0.9318
	Commercial vehicle	34,7%	17,8%	0.0000
Renewing customer indicator	No	19,2%	79,7%	3.7819
	Yes	31,3%	20,3%	0.0000
FCC record indicator	No	21,2%	99,9%	9.5254
	Yes	90,1%	0,1%	0.0000
Code of living mode	Non-owners	41,8%	17%	0.0000
	Owners	16,1%	83%	6.5557
PAN Leader	Good	20,2%	99,2%	8.1105
	Bad	78,5%	0,8%	0.0000
Share of maturity	Under 1,2%	4%	9,5%	4.0876
	Between 1,2 and 1,8%	17,6%	42%	2.0120
	Over 1,8%	27,9%	48,5%	0.0000
Percent contribution	Under 20%	25,2%	74,9%	0.0000
	Between 20 and 35%	18,3%	11,1%	3.1528
	Over 35%	3,3%	14%	13.1186
Product type	Traditional credits	23,1%	31,6%	0.0000
	Leasing contracts	26%	29,7%	0.9489
	Long-term rental	17,5%	38,7%	2.2043
Quality of the vehicle	Second hand vehicle	32,3%	17,3%	0.0000
	New vehicle	19,4%	82,7%	3.5649
Activity area	Farming	17,5%	4%	4.0279
	Others	23%	21,8%	2.1888
	Consumer goods	32,9%	2,4%	2.5779
	Construction industry	36,9%	12%	0.0000
	Retail business	26,5%	5,9%	2.6059
	Teaching and driving school	20%	11,8%	3.0798
	Hospitals	6,2%	20%	7.9172
	Hotels-Restaurants	45,2%	1,2%	1.1682
	Utilities	18,4%	18,8%	2.9888
	Transport	30,8%	2,1%	3.1995

6.2 Machine Learning Methods

With a Machine Learning algorithm, it is impossible to compute a Scoring Grid as we did with the Logistic Regression. Indeed, we do not have the predictor's coefficients associated. Hence, it is impossible to show some sort of Scoring Grid. However, for the Random Forest algorithm, it is possible to compute Variable Importance plots (VIMP). After training a Random Forest, it is natural to ask which variables have the most predictive power. Variables with high importance are drivers of the outcome and their values have a significant impact on the outcome values. By contrast, variables with low importance might be omitted from a model, making it simpler and faster to fit and predict. There are two measures of importance given for each variable in the random forest. The first measure is based on how much the accuracy decreases when the variable is excluded. This is further broken down by outcome class. The second measure is based on the decrease of Gini impurity when a variable is chosen to split a node. See this article for more information on Gini. One advantage of the Gini-based importance is that the Gini calculations are already performed during training, so minimal extra computation is required. A disadvantage is that splits are biased towards variables with many classes, which also biases the importance measure. Both methods may overstate the importance of correlated predictors. Neither measure is perfect, but viewing both together allows a comparison of the importance ranking of all variables across both measures. For further reading, see this paper and these slides[0].

6.2.1 Variable Importance - Random Forest

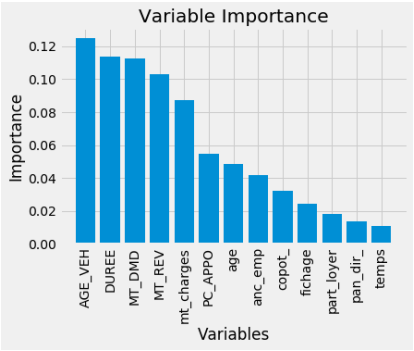

Figure 6.1: RAW dataset

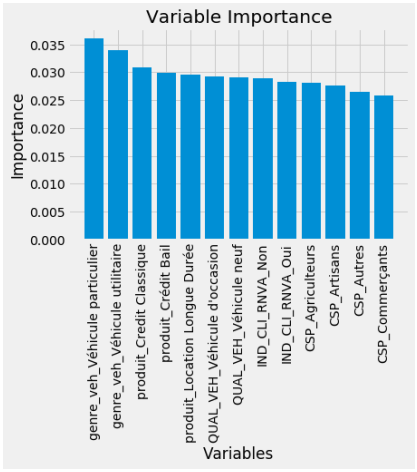

Figure 6.2: Subsample dataset

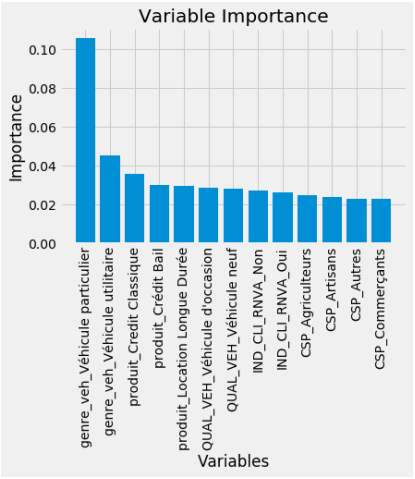

Figure 6.3: SMOTE dataset

We can clearly see that important variables are not the same for those 3 datasets. Indeed we clearly see a difference between RAW and Subsample/SMOTE Variable Importance's plots. When you look at the plot for the RAW dataset, you can clearly see that the biggest predictors are Customer age, Projected duration of financing & Amount of funding. Those seems really usual. However, when you look at the Variable Importance plots for the Subsample and SMOTE datasets, you can see that those 3 predictors do not appear. Indeed, the biggest predictors are Type of vehicle, Product type & Quality of the vehicle. Those latter are the same for both Subsample and SMOTE datasets, which seems normal as we created the SMOTE dataset from the Subsample one.

[0]Thanks to this article.

6.3 Metrics comparison

	Logit	Random Forest	SVM Lin	SVM Poly	SVM Sigmoid	SVM RBF
Gini Index	0.601	**0.607**	0.502	0.591	0.602	0.584
10/X Index	7.70%	**8.69%**	6.53%	7.62%	7.84%	8.20%

Figure 6.4: Results - RAW dataset

	Logit	Random Forest	SVM Lin	SVM Poly	SVM Sigmoid	SVM RBF
Gini Index	0.634	**0.675**	0.401	0.578	0.414	0.578
10/X Index	36.04%	**61.13%**	30.70%	45.59%	26.86%	42.4%

Figure 6.5: Results - Subsample dataset

	Logit	Random Forest	SVM Lin	SVM Poly	SVM Sigmoid	SVM RBF
Gini Index	0.766	**0.958**	0.761	0.845	0.656	0.859
10/X Index	79.13%	**99.61%**	81.00%	91.25%	73.04%	93.00%

Figure 6.6: Results - SMOTE dataset

Chapter 7

Conclusion

Achieving good data quality was the number one priority throughout this project. Indeed, the cleaning phase as well as the discretization were the two most important steps in our Project. In any data science project, the output quality depends on the quality of the input. The OSEMN Process was meaningful in order to make sure we had good data quality. The discretization impacted directly our results, which made that part a key step to a better understanding of our model.

Our project aimed at assessing the best possible Scoring Grid. However, the biggest problem in Scoring is to make sure we have a stable model. Ensuring stability over time in our model was really important. We had to correct some discretization made by our algorithm, hence loosing accuracy because some predictors were not stable over time and different modalities would cross each other.

Our results showed that the Random Forest outperformed every single other algorithm we tried (in terms of Gini and 10/X indices), from the classic (Logistic Regression) to the Support Vector Machine. Nevertheless, there is a trade-off when using Machine Learning techniques. In point of fact, there is a huge interpretability problem with Machine Learning algorithms. The Logistic Regression might not be the most accurate model, this is the only model with which we can produce a Scoring Grid. Random Forest might produce a Variable Importance plot, but as mentioned earlier, there are several articles saying that those plots are biased and that we should not trust those plots. Moreover, discretization is by far the most important step in a Scoring model. We saw that the way you discretizes predictors impact the outcomes a lot. Hence, making sure we get a good discretization algorithm could possibly be better than benchmarking Machine Learning algorithms versus classical methods.

In other words, we saw that Machine Learning algorithms tends to be more accurate than the classic model (Logistic Regression). However, the lack of interpretability and the complexity of those models make that the industry still uses Logistic Regression. The discretization makes a huge impact on the outcomes. Hence, most of the work should focus on the discretization algorithm.

Annexes

Annexes

Our codes are available on this GitHub repository.
The datasets needed to run this code can not be found as this is RCI Bank's property.

1 Python

Codes can be found on the `Python` folder.

WARNINGS
If you use Jupyter Notebook please use the .ipynb scripts in the Jupyter Notebook sub-folder.
You may need to install some libraries in order to run these codes.

1.1 Cramér's V

Please find the script here.

1.2 Random Forest

Please find the script here.

1.3 Support Vector Machine

Please find the script here.

2 R Studio

2.1 SMOTE

Please find the script here.

3 SAS (with Enterprise Guide)

3.1 Formats

Please find the script here.

3.2 Data Preparation

Please find the script here.

3.3 Data Exploration

Please find the scripts here and there .

3.4 Discretization

Please find the scripts here and there.

3.5 Logistic regression

Please find the script here.

3.6 Scoring grid

Please find the scripts here and there.

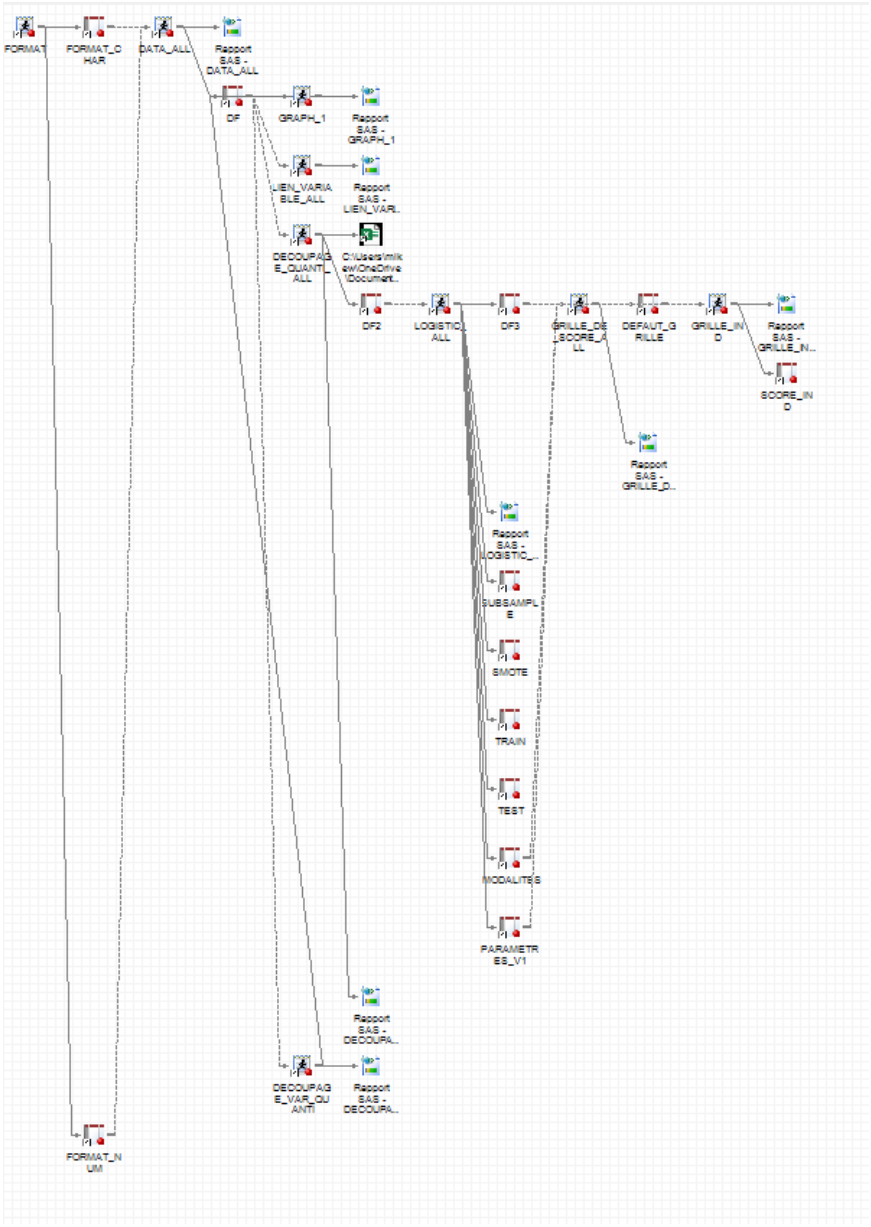

Figure 7.1: Project flow under Enterprise Guide

Bibliography

[1] H. Binder, O. Gefeller, M. Schmid, and A. Mayr. The evolution of boosting algorithms. *Methods of Information in Medicine*, 53(06):419–427, 2014.

[2] Leo Breiman. Random forests. *Mach. Learn.*, 45(1):5–32, October 2001.

[3] Tianqi Chen and Carlos Guestrin. XGBoost: A Scalable Tree Boosting System. *CoRR*, abs/1603.02754, 2016.

[4] Jerome H. Friedman. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29:1189–1232, 2000.

[5] Aurélien Géron. *Hands-on machine learning with Scikit-Learn and TensorFlow : concepts, tools, and techniques to build intelligent systems*. O'Reilly Media, Sebastopol, CA, 2017.

[6] Venu Gopal Lolla. Integrating Python and Base SAS. `<https://www.sas.com/content/dam/SAS/support/en/sas-global-forum-proceedings/2019/3548-2019.pdf>`, 2018. [Online; accessed 02-December-2019].

[7] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[8] T. Trevor Hastie, R. Tibshirani, and J. Friedman. The Elements of Statistical Learning. `<https://web.stanford.edu/~hastie/ElemStatLearn/>`, 2009. [Online; accessed 06-December-2019].