A picture containing drawing

Description automatically generated

Real-time credit scoring in FLOA Bank

by

**Youness ZAIM**

Under the guidance of

**Anais LAPEYRE** & **Christine THOMAS**

Toulouse School of Economics

Master 2 Econometrics & Statistics

Septembre 2021

Toulouse School of Economics

Real-time credit scoring in FLOA Bank

Summary

Within the framework of the Master 2 Econometrics and Statistics of Toulouse School of Economics, this report aims to review and assess methods for constructing a customized credit scoring model for online credit. We examine the process-flow involved in the scorecard modeling and highlight the probable causes of weak model performance. Such examination includes theoretical and empirical assessment of data quality, variable transformation and selection based on automated binning algorithms, statistical tests for model performances to assess its validity and practicality for implementation.

Acknowledgements

With boundless excitement and appreciation, I would like to express my sincere gratitude to the people who helped for the successful completion of my apprentice report:

* To my supervisor, Anais LAPEYRE, for hiring me as an intern and giving me the opportunity to do my apprenticeship in FLOA Bank, for the continuous support during my missions, for her generous advice, immense knowledge and encouragement.   
  I could not have imagined having a better mentor and role model.
* To my academic tutor, Christine THOMAS, for her guidance throughout the year and all members of University of Toulouse for the quality of their teaching and their daily efforts in guiding us towards a successful professional insertion.
* To my manager, Virginie LANGE, for her instructions and recommendations. Her cheerful and vibrant personality has always been my inspiration.
* To the beloved FLOA’s data science team:
  + Joanna MORAIS for all her kind support, genuine advice, and patience.
  + Vincent GALLMANN, for his generous support and guidance through the modelling process. His kindness will always be remembered and appreciated.
  + Minh-Tri NGUYEN for all the fond memories and kind support.

I would like to express my warmest thanks to all those who have contributed in all ways to the accomplishment of this work.

Contents

[I. Introduction 6](#_Toc80912197)

[1- Presentation 6](#_Toc80912198)

[2- Organization of FLOA Bank 8](#_Toc80912199)

[3- Data platform 9](#_Toc80912200)

[3.1 Data storage 9](#_Toc80912201)

[3.2 Data ingestion 11](#_Toc80912202)

[3.3 Exploitation and processing of the data 12](#_Toc80912203)

[II. Credit Scoring 15](#_Toc80912204)

[1- Types of credit scores 15](#_Toc80912205)

[2- Generic vs customized scoring models 16](#_Toc80912206)

[3- Coup de pouce UPCYCLE 16](#_Toc80912207)

[4- Monitoring machine learning models in production 21](#_Toc80912208)

[III. UPCYCLE credit score development 27](#_Toc80912209)

[1- Goal definition 27](#_Toc80912210)

[2- Data collection & preparation 28](#_Toc80912211)

[2.1 Data collection 28](#_Toc80912212)

[2.2 Data analysis 33](#_Toc80912213)

[3- Feature engineering 37](#_Toc80912214)

[3.1 Binning and weights of evidence 37](#_Toc80912215)

[3.2 Manuel binning 41](#_Toc80912216)

[3.3 Cluster Binning for Categorical Variables 42](#_Toc80912217)

[3.4 Time Features 43](#_Toc80912218)

[3.5 Hamming distance 44](#_Toc80912219)

[3.6 Impact coding 44](#_Toc80912220)

[3.7 Standard normalization 45](#_Toc80912221)

[3.8 Missing values 45](#_Toc80912222)

[IV. Modeling 46](#_Toc80912223)

[1- Predictive power 46](#_Toc80912224)

[2- Performance metrics 48](#_Toc80912225)

[3- Probabilities and lift 49](#_Toc80912226)

[4- Hyperparameter tuning 51](#_Toc80912227)

[5- Machine learning algorithms 51](#_Toc80912228)

[5.1 Random Forest 52](#_Toc80912229)

[5.2 Gradient Boosted Trees 52](#_Toc80912230)

[5.3 Logistic Regression 53](#_Toc80912231)

[6- Experiment results 54](#_Toc80912232)

[6.1 Confusion matrix and decision chart 54](#_Toc80912233)

[6.2 Lift curve 57](#_Toc80912234)

[6.3 ROC Curve 60](#_Toc80912235)

[6.4 Detailed metrics 63](#_Toc80912236)

[V. Interpretation 66](#_Toc80912237)

[1- Feature importance 66](#_Toc80912238)

[2- Partial dependence 67](#_Toc80912239)

[3- Subpopulation analysis 70](#_Toc80912240)

[4- Individual analysis 72](#_Toc80912241)

[VI. Conclusion 73](#_Toc80912242)

[VII. Appendix 74](#_Toc80912243)

[VIII. Bibliography 75](#_Toc80912244)

# Introduction

## Presentation

FLOA Bank is a French leader for web and mobile payment solutions by making consumers’ lives easier through payment facilities, instant loans and bank cards. FLOA is also partner to large e-tailers (Cdiscount, Oscaro, SFR, Vide dressing, etc.), key players in tourism (Selectour, Misterfly, Cdiscount Voyages, Pierre et Vacances, etc.), and fintechs (Lydia, Bankin’), for which it creates bespoke services. Its two retail banking brands are FLOA Bank for its B2C customers and FLOA Pay for its B2B partners. FLOA's products and services stand out, as they are easy to use for customers and quick to integrate for partners. FLOA boasts more than 3 million customers and grants more than €2 billion in loans for goods and services every year. In France, FLOA was named Best Customer Service Company 2021 (Service Client de l'Année 2021). With operations in France, Spain and Belgium, FLOA aims to become one of Europe’s leading providers of payment solutions.

In an e-commerce market experiencing very strong growth, associated with changing consumer habits and customer expectations with regard to payment methods, B**NP Paribas, Casino Group and Crédit Mutuel Alliance Fédérale signed an exclusivity agreement for the sale of FLOA to BNP Paribas by Casino Group and Crédit Mutuel Alliance Fédérale, setting up a strategic partnership between BNP Paribas and Casino Group. This** will enable FLOA to initiate a new development cycle, capitalizing on BNP Paribas' expertise and areas of business, particularly in view of European deployment.

FLOA Bank presents a range of innovative products divided into 4 main categories:

* Online credit: Personal loan, revolving credit and credit insurance. More information on all the online credits: https://www.floabank.fr/credits/gamme-credits
* Bank cards: Casino bank card, Cdiscount MasterCard, gold card. More info on all bank cards: https://www.floabank.fr/mastercard/gamme-cartes
* Insurances: Credit and cards, family and leisure, health and providence. More info on all insurances: https://www.floabank.fr/assurances/gamme-assurances
* Payment facilities: Payment in several instalments (CB3X, CB4X and CB10X) on partner sites, instant mini-loans also called "Coup de pouce" (directly or via partnerships such as Lydia, Bankin, Cdiscount).
  + The "Coup de Pouce" credit allows Internet users to benefit from a loan repayable in less than 90 days in 3 or 4 times (3 or 4 monthly payments), with a minimum of administrative procedures and a minimum of supporting documents. This credit offer is characterized by the flexibility of managing payments with digressive fees according to the amount of the Helping Hand. They vary from 1.48% to a maximum of 3.13%. In practice:
    - For a ‘Coup de Pouce’ in 3 instalments: You pay the costs of your loan on the day of your application and then pay the total amount borrowed in 3 constant instalments over a period of 90 days.

R1 = Credit fees on the day of the order D

R2 = 2nd instalment (1/3 of the amount requested) => D + 30 days

R3 = 3rd instalment (1/3 of the amount requested) => D + 60 days

R4 = 4th instalment (1/3 of the amount requested) => D + 90 days

* + - For a ‘Coup de Pouce’ in 4 instalments: You pay the costs of your credit on the day of your request and then you pay the total amount borrowed in 4 constant instalments over a period of 90 days.

R1 = Credit charges + 1st instalment (1/4 of the amount requested) on the day of the order D   
R2 = 2nd instalment (1/4 of the amount requested) => D + 30 days

R3 = 3rd instalment (1/4 of the amount requested) => D + 60 days

R4 = 4th instalment (1/4 of the amount requested) => D + 90 days

An example to illustrate how the "Coup de Pouce " works: If your mini-loan is 600 €, you pay an amount of 18,78 €, the day of the order. Then, you pay a constant monthly payment of 200 € during the next 3 months. In total, your credit costs 618.78 €. More information on Coup de Pouce :https://www.moncoupdepouce.com/

* + Payment in instalments (Le « paiement en plusieurs fois ») allows Internet users who are customers of FLOA’s partner merchant site to pay for their purchases in one, three or four instalments (one, three or four monthly instalments) with their bank card. The Customer's repayment schedule will begin on the day the order is validated and will be spread out as follows:
    - For the "1 X deferred" and "1 X deferred FREE" Payment Service:

R1 = 1st instalment on D + freely chosen period of time up to a maximum of 30 days.

* + - For the "3 X" and "3 X FREE" payment service:

R1 = 1st due date on D R2 = 2nd due date => D + 30 days   
R3 = 3rd due date => D+ 60 days

* + - For the "3 X deferred" payment service:

R1 = 1st due date on D + 30 days  
 R2 = 2nd due date => D+ 60 days   
 R3 = 3rd due date => D + 90 days

* + - For the "4 X" and "4 X FREE" payment service:

R1 = 1st due date on D   
 R2 = 2nd due date => D + 30 days  
 R3 = 3rd due date => D + 60 days  
 R4 = 4th due date => D + 90 days

An example for a purchase on one of FLOA’s partner sites for an amount of 1500€. If you use the CB3X without fees proposed by FLOA Bank, a contribution corresponding to one third of the order will be requested on the day of the order. Two monthly payments corresponding to one third of the order will be taken 30 and 60 days later. No management or file fees: More information on all payment facilities: https://www.floabank.fr/solution-paiement/gamme-solution-paiement

## Organization of FLOA Bank

The Data Department in FLOA Bank is composed of four teams:

* Data Factory under the responsibility of Benjamin Laurent: The team is in charge of feeding all data (partners, back & front IS, repositories, etc.) on the data platform (Snowflake), industrializing data flows (Talend) and setting up an enterprise data model.
* BI & Data Analytics under the responsibility of Zouheir Azahaf: The team is in charge of the industrialization of reporting in Tableau, BI analysis for the different departments of the company, and support for the business lines on their data challenges, especially on the specification of data marts.
* Data Science under the responsibility of Virginie Lange: The team is in charge of all predictive modeling for the company (risk, fraud, collection, finance, marketing, etc.) and in particular the implementation of real-time scoring on our customer paths.
* Strategy & Data Projects under the responsibility of François Le Nouvel: The team isresponsible for the coordination of Data subjects and transversal operating modes as well as the internal/external visibility of data at FLOA.

The Data Department is also part of the FLOA governance with the use of Jira to help gain in efficiency, rigor, visibility, traceability and management.

The chart below illustrates the FLOA organization:

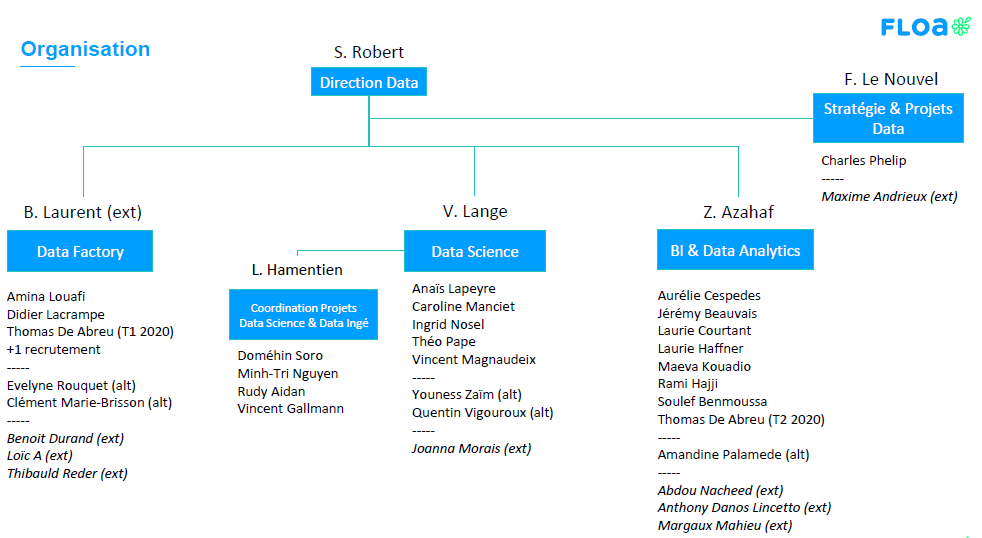


Figure 1: Diagram of the organization of FLOA

## Data platform

The Data platform at FLOA brings together a set of tools and services to store (via Blob Storage, Datalake, Snowflake), process/prepare (via Talend or Dataiku), govern (via DataGalaxy), visualize (via Tableau), analyze (via Tableau or Dataiku) and model (via Dataiku) data.

### Data storage

* The Azure Blob Storage: serves as a buffer zone for storing raw data and as a gateway to the Data platform (SI FLOA, Euro Information, Eureka, etc.). Data arriving on the blob is not intended to remain there but to be automatically migrated to Snowflake (or the Datalake in certain cases)
* Data Lake: allows to store in file mode a very large quantity of data. These data can be stored in any format, in a raw way or after a reworking. They can be kept for an indefinite period of time (within the limits of RGPD). Historically, datalakes appeared in the so-called "Big Data" architectures in order to allow the analysis of huge volumes of data at a lower cost than storage on a classic database. In the meantime, databases have evolved and their cost has decreased. In addition, managing data in file mode on a Datalake has often appeared tedious and therefore costly for standard uses. This is why in FLOA, the Datalake, provided by Microsoft Azure, and is intended to be used for the storage and use of unstructured data (MongoDB) because they cannot be stored in database format (Snowflake). These data must be regularly purged to meet the RGPD rules.
* Snowflake (or SNF): is a SQL data storage and analysis service offered as a SaaS software (hosted in the Azure cloud in the case of FLOA). It is the first data warehouse designed for the cloud. Its multi-cluster architecture separates data storage into three distinct functions: storage; computation (virtual warehouses); services (user authentication, management, security, query optimization).

The tree structure in Snowflake is as follows:

* Database
  + Schema(s)
    - Table(s)
    - View(s)

In FLOA, the databases correspond to the different levels of storage (ODS, DWH, DTM) but also to environments dedicated to the business (BIDS, FRAUDE, etc.). Each database will have a production version (e.g. DWH\_PRD), a test version (e.g. DWH\_REC) and a development version (e.g. DWH\_DEV). For the schemas, it will be a logical grouping allowing to target more quickly the stored data (Ex: MASTER, FILE, CASHBACK, SCORE\_ERK). The table names correspond to those defined when modeling the ODS, DWH and DTM layers.

Snowflake is a tool whose use is intended for Data teams (DataOps, Data Science and Data Analysis):

* Snowflake - OPERATIONAL DATA STORE (ODS): This storage phase concerns structured and semi-structured data. The ODS allows to centralize in our Snowflake database all the heterogeneous data entering the Data platform (AR, external partners, open data, etc.) This storage space allows to standardize the storage format (Snowflake) but also the format of some types of fields (Date, Number, etc.). For an input data source, a target data source is fed into the ODS. It is therefore a raw data storage with a first level of technical harmonization. This level of storage is intended to be fed by the DataOps team and used by the DataOps and potentially Data Science teams.
* Snowflake - DATA WAREHOUSE (DWH): The DATA WAREHOUSE succeeds the ODS layer in Snowflake. It allows to arrange, group, consolidate the data stored in the ODS in a model corresponding to the activity of the company. These tables contain information at the finest granularity with few calculated indicators. We can talk about a tabular view of data grouped by functional family (e.g.: Instruction table in the PnF universe, Stakeholder table in the Cards&Credit universe, etc.) This level of storage is intended to be fed by the DataOps team and used by the DataOps and Data Science teams and all the company's Data Analysts.
* Snowflake - DATAMARTS (DTM): The DATAMARTS succeed the DWH layer in Snowflake. Each DATAMART corresponds to a precise business need/use. There are many of them within the company. The DATAMART will allow to consolidate the DWH layer by gathering the relevant information present in several DWH tables and by storing them in a single table. DATAMART allows a more efficient and simpler use of the data available in the DWH. The business gets what it needs, stored the way it wants and with more efficient response time performance. This level of storage is intended to be fed by the DataOps teams and used by all DATA users within FLOA.

### Data ingestion

#### Tools used for data ingestion

The data ingestion phase consists of retrieving data from the blob storage that is stored externally, either in the FLOA SI or at Euro Information or at our partners or via open data. The data can be retrieved

* Either by push (the data provider pushes the data to the blob storage)
* Or by connecting directly to the database or to an intermediate SFTP (via Talend connectors)
* Or by using a dedicated query tool, such as the Focus / Webfocus tools with Euro Information

Different tools to support this data ingestion phase:

* Talend: Talend is an ETL (Extract, Transform, Load) that collects data from several sources, structures it, centralizes it, and prepares it to make it available to users. Talend's tool is used exclusively by the DataOps team to make prepared data flows available to the company
* Focus / Webfocus (outside the Data platform): Focus (older technology) and Webfocus (newer technology) are tools that code queries to extract BCA data hosted in the Euro Information environment (SIDU). Eventually, the goal is to get rid of these queries by directly deporting the data uses on the platform
* DataGalaxy: DataGalaxy is a web & collaborative data mapping tool, essential to comply with the RGPD. The tool will eventually allow to have an exhaustive view of metadata and data processing on the Data platform, by offering search, impact analysis and data lineage functionalities. DataGalaxy also allows tagging some metadata and these tags can be used for example for the criticality level of personal data in the framework of the RGPD. DataGalaxy is a collaborative tool that will be open to all (not yet available - the opening will be done progressively) with 3 types of licenses depending on the use (steward, explorer, viewer)

To better understand the "journey of a data", here is an example of lineage for an EI table arriving on the platform via the Blob which is ingested (via Talend) into Snowflake's ODS and DWH layers, on which the DTM layer is based.

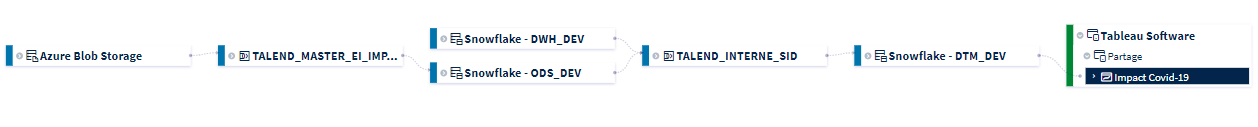


Figure 2: Modeling the path of the data

#### Processing of the raw data during ingestion

The data preparation includes the different steps between the ingestion of the data and its availability to the user and can be summarized in the following steps:

* Retrieve the data from the blob storage,
* Clean the data and control the quality of the data,
* Structure the data (according to the data model being developed),
* Enrich the data by crossing it with data from the same or another table,
* Publish the data in Snowflake or in the Datalake

The use of the data can be in real time:

* In FLOA, our product subscription paths query production databases to store data (to retain information related to the customer's request) and to retrieve data (e.g. customer recognition) in real time.
* Other usages include calling the credit score, fraud detection with the Threatmetrix tool (TMX), identity verification with Netheos, etc.

On the other hand, the use of data can be offline and corresponds to a usage of business monitoring or steering reports, data analysis, score modeling, marketing campaign creation, etc.

### Exploitation and processing of the data

FLOA bank provides different tools to accompany this data exploitation phase:

* Dataiku (or DSS): DSS is a Data Science tool that allows not only the analytical processing of data but also different types of modeling (prediction score, appetence, segmentations etc.) while offering a collaborative approach. The platform is composed of different environments and allows to prepare, model (Design Node environment), automate and historize the workflow (Automation Node environment), and deploy the production (API Deployer and API Node environments). For more information on the technical environment around Dataiku, please refer to:

https://www.dataiku.com/

The use cases of Dataiku DSS are numerous. The platform can be used for scoring, marketing / appetence analysis, fraud detection, data management, demand forecasting, etc. Dataiku brings us the possibility:

* + To work on more data (in terms of volume and variety),
  + To do more "modern" modeling with the use of new methods like neural networks,
  + To be more agile in the evolution of scores thanks to machine learning, the objective being always to play with the acceptance/risk trade-off.

Dataiku is a tool that is intended for use by data scientists and accounts for:

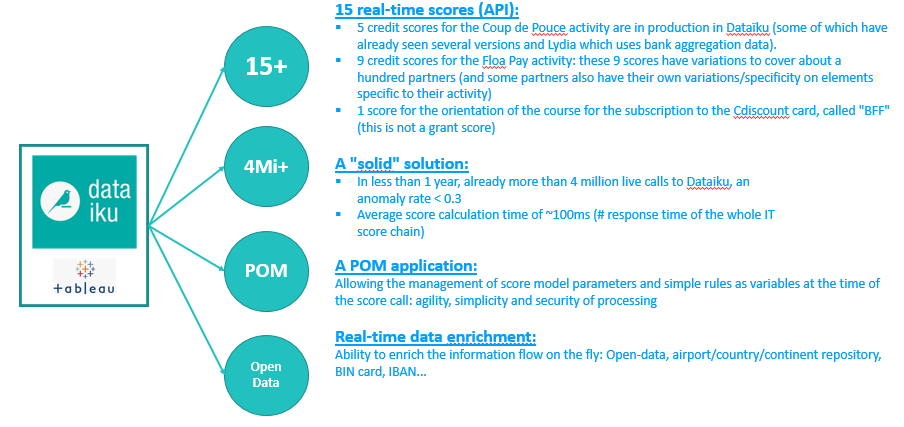


Figure 3: Dataiku usage in FLOA

* Tableau: Tableau is the tool that has been selected by FLOA bank in the data platform as a data visualization tool (DataViz). DataViz refers to all visual representations of data in order to help the user better understand complex and numerous information or data and to ensure a better understanding of the data and the issues related to it. Tableau is one of the world leaders in the market with Power BI (Microsoft) and QlikSense (QlikView). It uses a file structure of workbooks and sheets similar to Microsoft Excel. A workbook contains sheets. A sheet can be a spreadsheet (also called a view), a dashboard or a story.

There are 3 types of licenses to access Tableau depending on the use - the tool is intended to be open to all:

* + Creator: for advanced uses; this status allows to connect to any data source outside the data platform
  + Explorer: allows you to create your own workbook from enterprise data made available on the data platform
  + Viewer: allows you to access the visualizations created by Creators & Explorers and to customize the visualization according to your own needs (alerts, views, etc.)

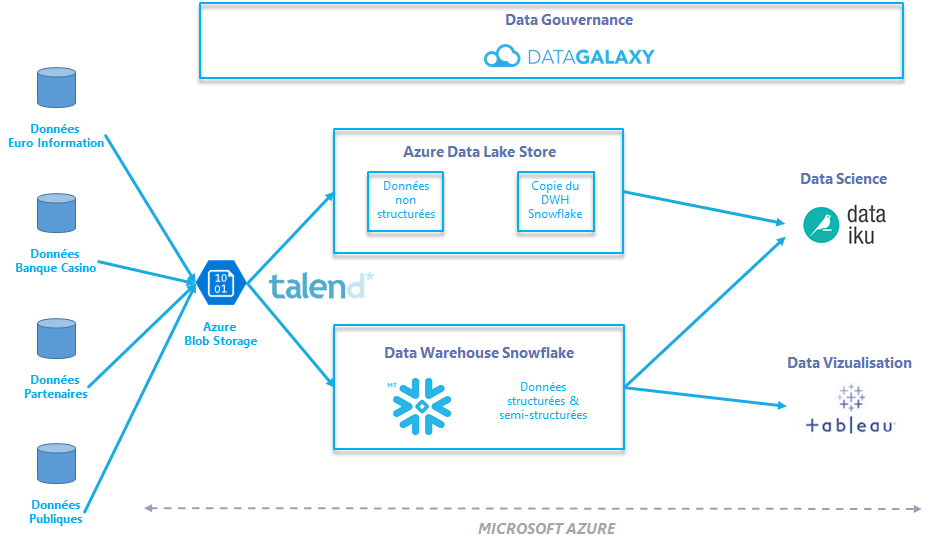


Figure 4: Data platform in FLOA

# Credit Scoring

## Types of credit scores

Scoring is one of FLOA's main activities. Based on quantitative and qualitative data, such as a user's behavior on a website or the frequency and value of his purchases, scoring consists of assigning a score to determine the probability of response to an offer or the risk of non-payment.

The Data Scientists use different types of scores:

* Credit score or risk score: deciding from data derived from the client’s application and public data, whether FLOA accepts or rejects the order based on the risk estimate which the probability of default. These scores are all intended to be internalized in the Dataiku tool.
* Credit default recovery score: to evaluate whether it is more relevant to propose an automatic "self care" path or to go through an advisor for a person entering into a credit default recovery. The objective here is to optimize the contact strategy according to the past behavior of customers in the same group to restore their unpaid situation. The right treatment for each profile will allow FLOA to save money and focus its efforts on those most in need of further treatment.
* Fraud score: detect (a priori or a posteriori) if a request seems to correspond to a fraud.
* Appetence score: used in direct marketing and which reflects the probability that a prospect will become a consumer of the product or service being promoted or that he will respond favorably to the offer made to him.
* Attrition score: used in customer portfolio management and translates the probability that a customer or subscriber will switch to a competitor or terminate his contract.

A score is first modeled with a learning database population to answer a given problem and validated by the achievement of one or more KPIs (e.g.: acceptance rate and risk rate for a grant score). This score can then be used:

* Real time scoring: Real-time scoring involves using the model to make predictions on individual rows of data via an API.
* Batch scoring: Batch scoring involves using the model to make predictions on multiple rows of data at a given frequency.

In addition to scoring activities, the automation of decisional reporting has many advantages in the face of the multiplication of needs and allows for significant time savings on a daily basis. At the request of the various Data Science/Finance/Risk and Fraud teams, the role of the analysts is to create dashboards with precise KPIs into which the data will be automatically integrated. These dashboards are then published on Tableau Server as activity monitoring.

Some examples:

* Cards and Credits universe: Operational monitoring of collections, acceptance/refusal rates, KPI's Granting, KPI's Risk.
* PNF Universe: Monitoring of Pnf and Coup de Pouce scores in terms of acceptance and breakage (unpaid).
* Marketing and customer knowledge: Customer activity monitoring, campaign assessment.

## Generic vs customized scoring models

In FLOA Bank, there are two types of credit scoring models:

* Generic credit scoring model: based on an overall behavior of different customers. The goal is to have a general model that aims to rank order the risk appropriately and simultaneously define the borrower characteristics, for a specific product’s offering, that best predict future behavior in the repayment of the loan.
* Customized credit scoring model: based on a particular population and the outcome of the lending decisions that captures the behavioral and other characteristics of the subprime population targeted.

In order for the customized scoring to be meaningful and aligned to the needs, it is necessary that there is sufficient information on the credit lending product.

## Coup de pouce UPCYCLE

Coup de Pouce UPCYCLE is one of several Coup de Pouce offers in FLOA. Based on Coup de Pouce production in first trimester of 2021, UPCYCLE presents 3.5% of total production behind Coup de Pouce Cdiscount (65.5%), Direct (19.5%) and Lydia (10.2%). In the other hand, UPCYCLE default cases presents 6.7% of the overall default’s cases across different Coup de Pouce providers.

It aims at proposing an alternative offer, which corresponds to 500€ or 1000€, to refusals of large amounts ranging from 5.000€ to 15.000€ (PPR / PPC). The UPCYCLE idea intent at testing the potential customers on a new product like Coup de Pouce, which is based on smaller amounts, and then possibly cross sell or potentially let them in for a large credit. Coup de Pouce UPCYCLE scoring model is in production since December 2019. It is a generic credit scoring model based on customer’s behavior from different Coup de Pouce partners.

Generally, generic credit scoring models do not accurately predict the level of the risk. Thus, banks that use generic models should not assume that their loss rates will be the same as those reflected in the industry odds charts. How accounts ultimately perform depends on a number of factors, including account management techniques used, the size of loan granted, and so forth. However, the development and implementation of scoring models and the review of these models present inherent challenges.

In that context, my first mission in FLOA Bank aimed at creating a new Tableau dashboard that allow the risk management unit as well as data scientists to monitor the production of the scoring models from a risk perspective and to simulate the future default rates.

Using different KPIs, we analyze the production behavior and the robustness of the score model based on:

* The distribution of orders by profile (UNKNOWN/TRUSTED/WHITELIST detailed in following chapters) and offers (3X/4X) over time.
* The distribution of orders by scores over time.
* The average basket per score over time.
* The default rate by score: theoretical rate vs observed rate
* The simulation of default rate for the next 6 weeks.

From the Tableau dashboard, we illustrate the scoring model behavior on all customer’s profile of Coup de Pouce UPCYCLE. The chart below illustrates that the generic score model for Upcycle is not properly distinguishing the bad from the good payers. The default rates, as the percentage of unpaid loans i.e Coup de Pouce, varies widely across the risk scores and does not follow the theoretical trend of the model. On one hand, we notice higher default rate on good scores: default rate of 12% on score of 20 vs default rate of 10% on score of 14 and it reaches 15% on score of 4 but rises to 17% rate on score of 12. On the other hand, the default rate for each score should virtually be the same or close in accordance with the model in theory as the higher the score, the lower the credit risk. However, on score of 20, the model simulates 1.3% theoretical default rate whereas, in production, the default rate reaches 13%.   
Finally, regardless of the risk score, the average default rate remains very high close to 14% comparing to other Coup de Pouce partners with a customized scoring model.

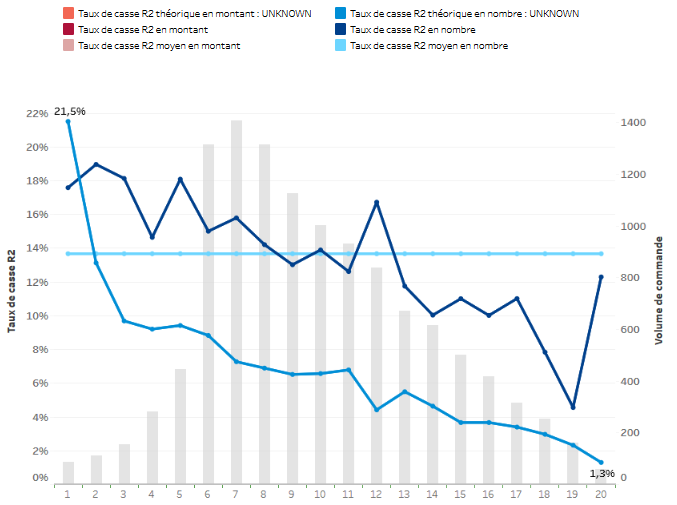


Figure 4: Default rate by score for UPCYCLE

In order to illustrate the performance of a customized scoring model, we take as an example the credit score model of Coup de Pouce CDISCOUNT, one of FLOA largest Coup de Pouce providers.

We notice that the risk level across the risk scores is decreasing alongside the theoretical default rate. The volume of orders is equally distributed according to the risk level: Higher volume on good scores beneath the average default rate at 5% and lower volume on bad scores with default rates slightly above the average but remain below the theoretical risk in each score. And regardless of the risk score, the average default rate remains close to 5%.

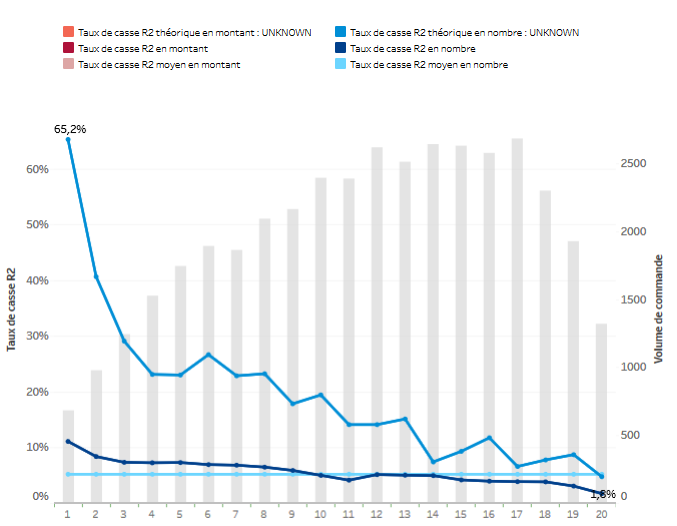


Figure 5: Default rate by score for CDISCOUNT

Furthermore, the following chart displays the overall trend of the default rate varying over a wide range from 7% to 23%. The projection of the default rate on the last weeks of production, according to different empirical assumptions, indicates a continuous trend of the default.



Figure 6: Historical and projected default rate for UPCYCLE

In addition, the following chart describes the proportion of Coup de Pouce Upcycle orders by score ranging from 1 (High risk) to 20 (Low risk). We see that most of the orders are on low score, from 1 to 5, meaning that the Upcycle population is risky.

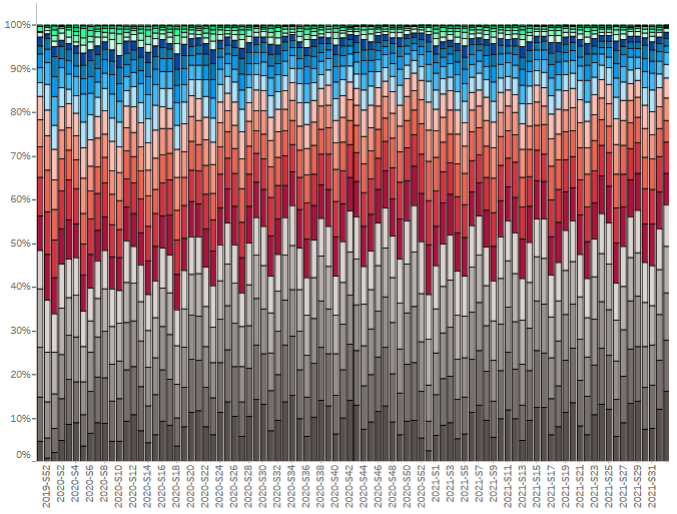


Figure 7: Distribution of Coup de Pouce orders by score for UPCYCLE

From this analysis, regarding the evaluation and the ongoing monitoring, we deduce that the population of Coup de Pouce Upcycle is risky and that we may identify concerns about the generic model performance of Coup de Pouce UPCYCLE.

## Monitoring machine learning models in production

Determining the perfect time to upgrade a generic model can be challenging. The Tableau dashboard of risk monitoring can help to illustrate the decision making by comparing the theoretical model results and the risk level of the model in production. The default rate projection for the last weeks of production can also be a good indicator for the expected default rate.

In addition, once we reached more than 1.000 default cases on a product, the management can feel confident that it may be the time to challenge the generic model with a customized one. In our case, for Upcycle scoring model, until July first, the default reached 1567 cases which corresponds to 1M€ loss. The total orders attained 11.500 which corresponds to 7.6M€.

Switching to a specific model brings good benefits regarding having a closer look to the data population and automating the model lifecycle ensuring that the data as well as the models are up to date.

Monitoring Customized credit scoring models in production is an important, but often tedious task for data scientists. Conventional model retraining strategies are often based on monitoring model metrics. However, for many use cases, monitoring model metrics, such as AUC, may be insufficient. This may be particularly true in cases where long periods of time pass before receiving feedback on the accuracy of the model’s predictions. Over time, a machine Learning model gradually loses its predictive power. This is commonly known as "drift". If this phenomenon is not detected quickly, it can have a significant impact on the performance of our models. It is therefore important to set up a loop that allows us to refresh our models when they degrade too much.

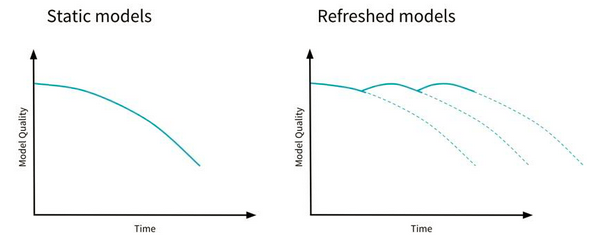


Figure 6: Model quality of static models vs refreshed models

In general, such deviation may occur for two possible reasons: Concept Drift and Data Drift.

* Concept drift corresponds to a change in the statistical properties of the target variable.
* Data drift, on the other hand, corresponds to a change in the statistical properties of the explanatory variables.

In FLOA Bank, data scientists define several possible strategies:

* Re-train the customized model on more recent data and monitor the performance indicators (AUC, Lift, F1-score, ...). If these indicators drop by a certain percentage, it is a sign that there is a drift in our data. Although this methodology may be correct, changing a model on production based on the performance metrics is risky. It may not go well as planned as we usually have to wait several days or even several months to see if the event has taken place.
* Monitor the evolution of the statistical properties of the variables can be more useful by looking at the recent data the model has had to score, and statistically compare it with the data on which the model was evaluated in the training phase. If these datasets are too different, the model may need to be retrained.

Dataiku developed a plugin used by the data science team called “model drift monitoring” which aims at providing views to analyze the potential drift of machine learning models. With this plugin, we can measure Drift by examining if new data waiting to be scored has diverged from the training data. If this new data is significantly different from the data used to train the model, it is likely that the model will no longer perform well, and may need to be retrained.

To detect drift between the original test dataset used by the model and the new data, the plugin stacks the two datasets and train a RandomForest (RF) classifier that aims at predicting data’s origin. This is called the domain classifier in the literature.

To have a balanced train set for this RF model, we take the first n samples from the original dataset and the new one, where n is the size of the smallest dataset.

If this model is successful in its task, it implies that data used at training time and new data can be distinguished, we can then say that the new data has drifted.

The plugin provides a new model view with the following visual elements:

* + - Drift score:

The plugin applies a binomial test on the accuracy of the domain classifier. The null hypothesis being there is no drift. There are two types of information that are computed:

* The p-value which gives the probability of having at least the computed accuracy under the null hypothesis.
* The confidence interval of the computed accuracy given the sample size.

The confidence interval gives us an index about the reliability of the drift score. If we get an accuracy score of 0.65 ± 0.2, the score is not very reliable as it overlaps with 0.5, which means no drift.

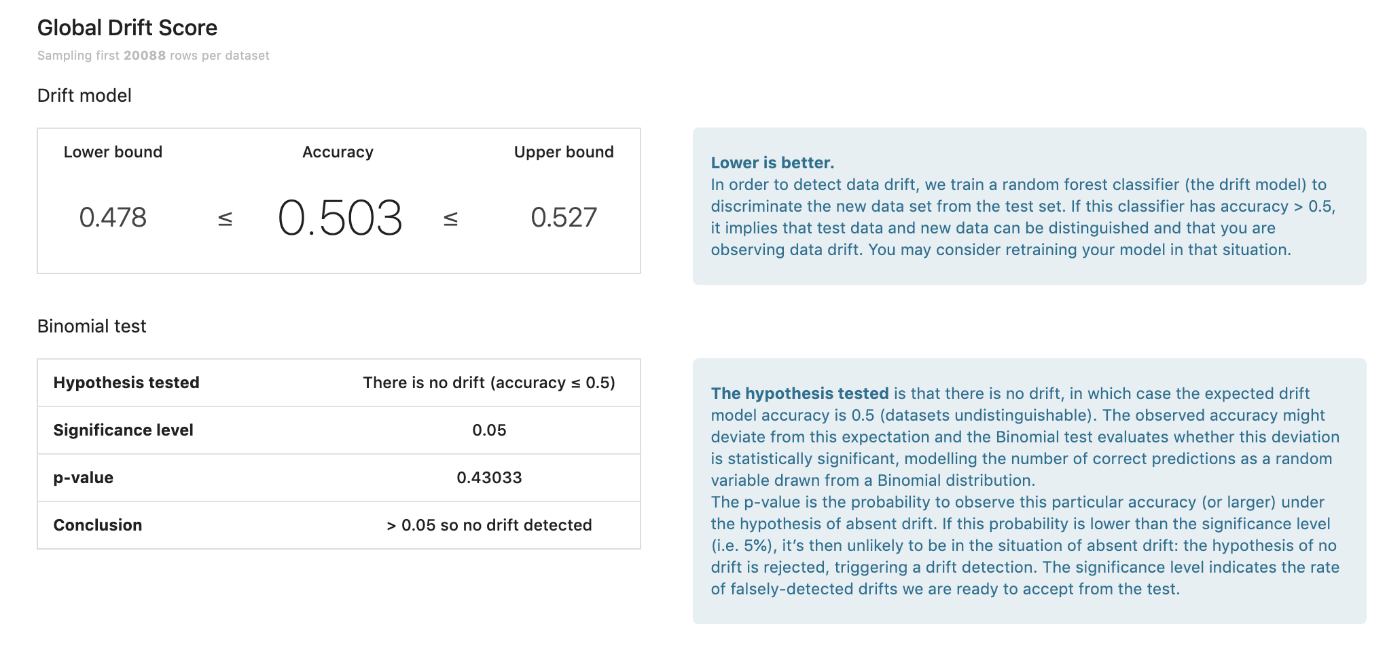


Figure 7: Drift score

* + - Fugacity:

In addition to a drift score, the plugin provides a number of other insights. The Fugacity table expresses the difference between expected and observed samples. It compares the proportion of samples predicted in each class when applying the model on both the test and the input datasets.

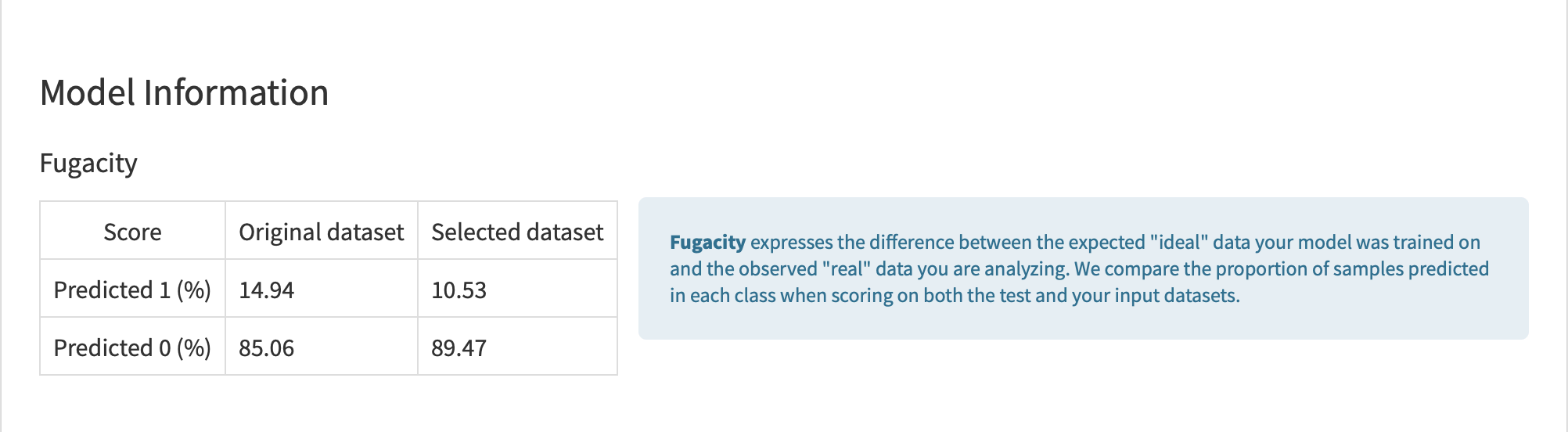


Figure 8: Fugacity table

* + - Density chart:

The density chart is the probability density estimation for a given prediction class when scoring both the test dataset and the new input dataset. Visually different probability density estimations indicate high data drift. The inverse is not true, as we can have a high drift score but a similar predicted probability density estimation. One such situation is when we have high drift of a feature that is not important in the original model.

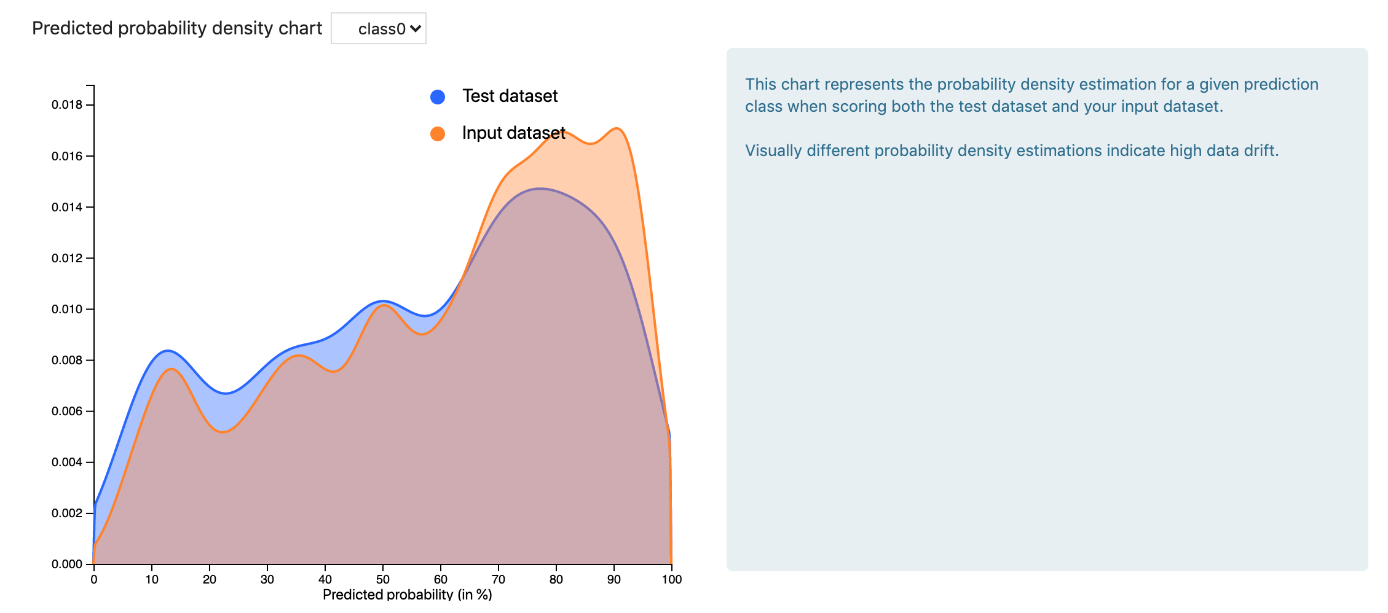


Figure 9: Density chart

* + - Scatter plot:

It is important to consider the data drift score in the context of the importance of certain features. For example, if the data drift score is high, but the features responsible for the drift are of low importance, the behavior of the model may remain the same.

The scatter plot shows feature importance for the original model on the x-axis versus feature importance for the (data classifying) drift model on the y-axis. Features in the top right quadrant of this scatter plot are highly drifted (i.e. they are powerful in distinguishing test data from new observations), but also of high importance for the original model. In this situation, you can expect the performance of the model to degrade as the model does not apply to your new observations.

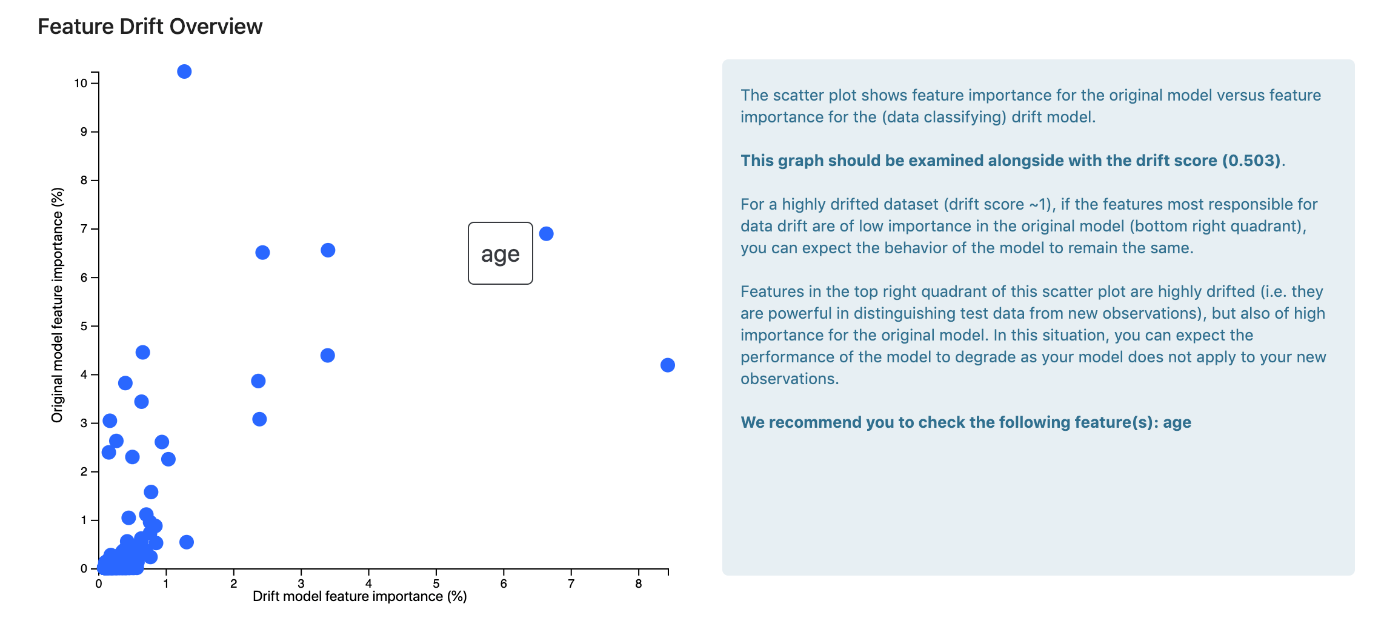


Figure 10: Scatter plot for feature drift

Second, by computing data drift between two dataset that have the same schema. The goal is to evaluate the same population, using the same KPIs, between different time stages. This recipe aims at data scientist who mostly build scoring models from data for a customized scoring model, each day when new data comes in they want to have a tool to assess whether or not the new data is “different”. It provides 2 types of information:

* Drift score: Data scientists can create a custom check with their own threshold and put these in a scenario for drift monitoring.
* Most drifted features: if the drift score is the threshold defined, we can define the features behind this drift.

Our thanks go to Vincent GALLMANN, senior data scientist in FLOA and one of Dataiku power users, for helping us to understand these technics as well as providing its different illustrations in Dataiku DSS as below:

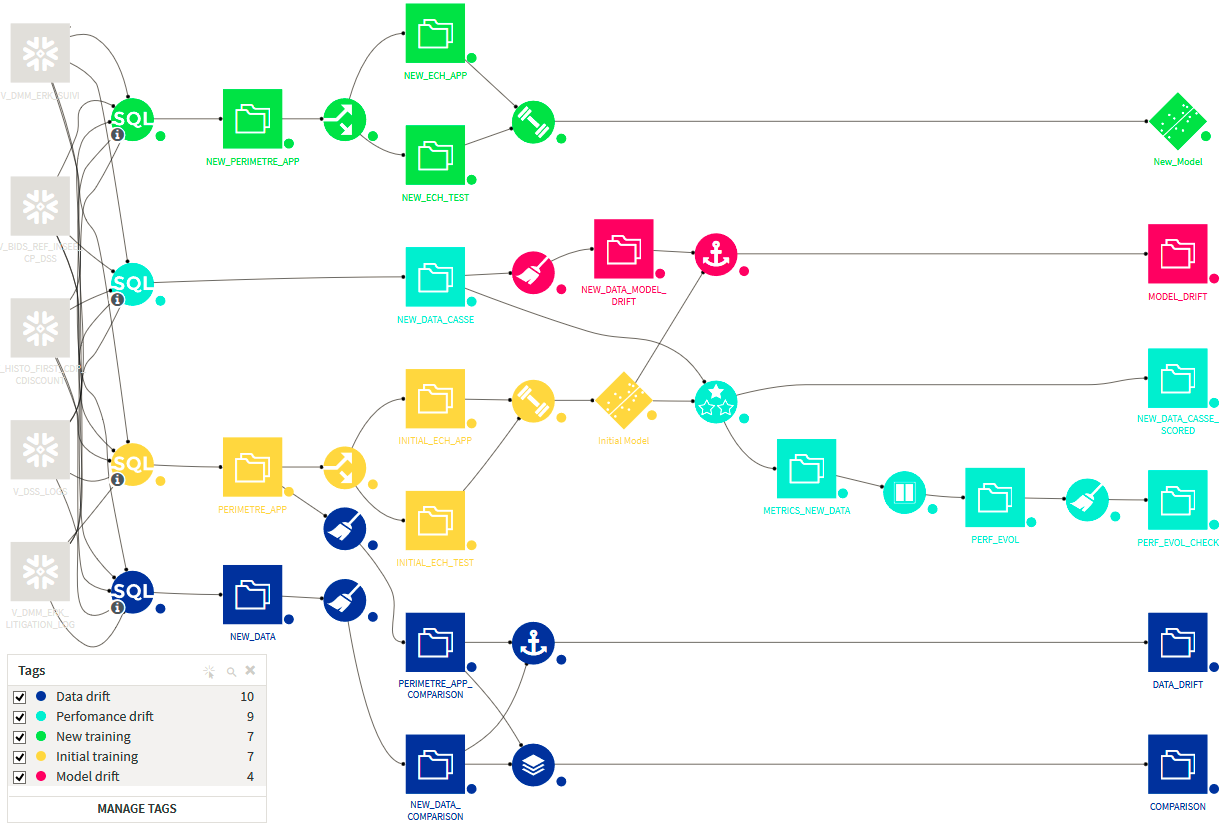


Figure 11: Dataiku flow for drift application

In conclusion, when properly designed, customized models are usually more reliable than generic subjective or judgmental methods. However, development and implementation of scoring models and review of these models present inherent challenges. These models will never be perfectly right and are only good if users understand them completely. Further, errors in model construction can lead to inaccurate scoring and consequently to booking riskier accounts than intended and/or to a failure to properly identify and address heightened credit risk within the loan portfolio. Errors in construction can range from basic formula errors to sample-bias to use of inaccurate and erroneous data. The impact on the score could be substantial.

# UPCYCLE credit score development

## Goal definition

A credit scoring project starts with understanding the business objectives. With the assistance of Anais LAPEYRE, we had several exchanges with the risk and commercial teams to highlight a new plan for Upcycle scoring model. Alongside the risk team and after defining the new budget starting from June 2021, we set a 21% theoretical default and an approval rate of 70%. On the other hand, the marketing team brought insights about the user’s types, the origins of the orders and the business strategy. Hence, the new scoring model will cover:

* Refusals of FLOA personal loans up to 15.000 € (PPRTOCDP): It covers the major part of Upcycle clients since its launch.
* Refusals of FLOA personal loans ranging from 7.000 € to 10.000 € (PPCTOCDP).
* Business providers (Refus PPR Leadin): LEADIN.
* FLOA credit card: The Upcycle model is set to score potential clients for the credit card Casino-Cdiscount starting late August 2021.

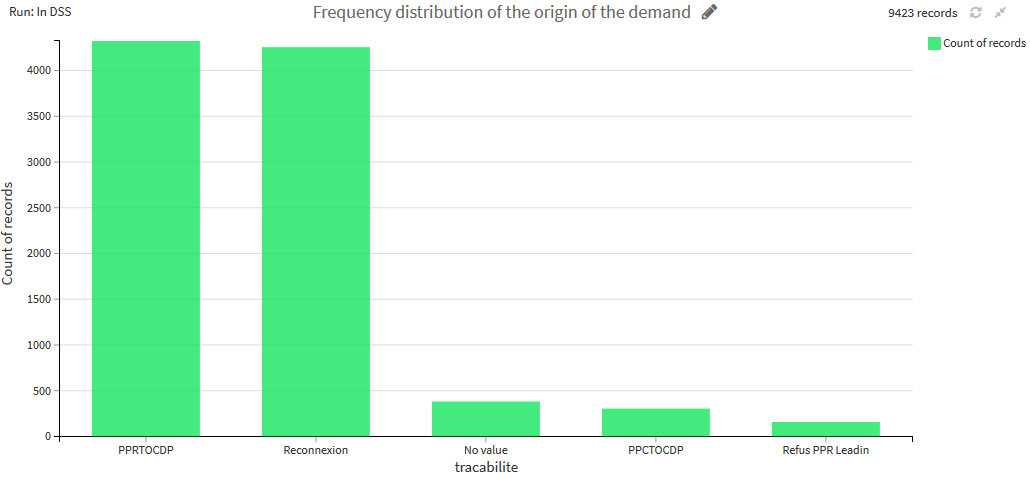


Figure 12: Order’s origin of Coup de Pouce UPCYCLE

*\*Reconnexion modality refers following PPC or PPR request but whose origin has been lost*

Once the scope of the business is defined, the machine learning engineering part starts. The figure below illustrates the following stages of a scoring project life cycle and consists of: 1) goal definition, 2) data collection & preparation, 3) feature engineering, 4) model training, 5) model evaluation, 6) model deployment, 7) model serving, 8) model monitoring, 9) model maintenance. At some stages (indicated by dashed arrows), a decision can be made to go back in the process and either collect more data or collect different data and revise features.

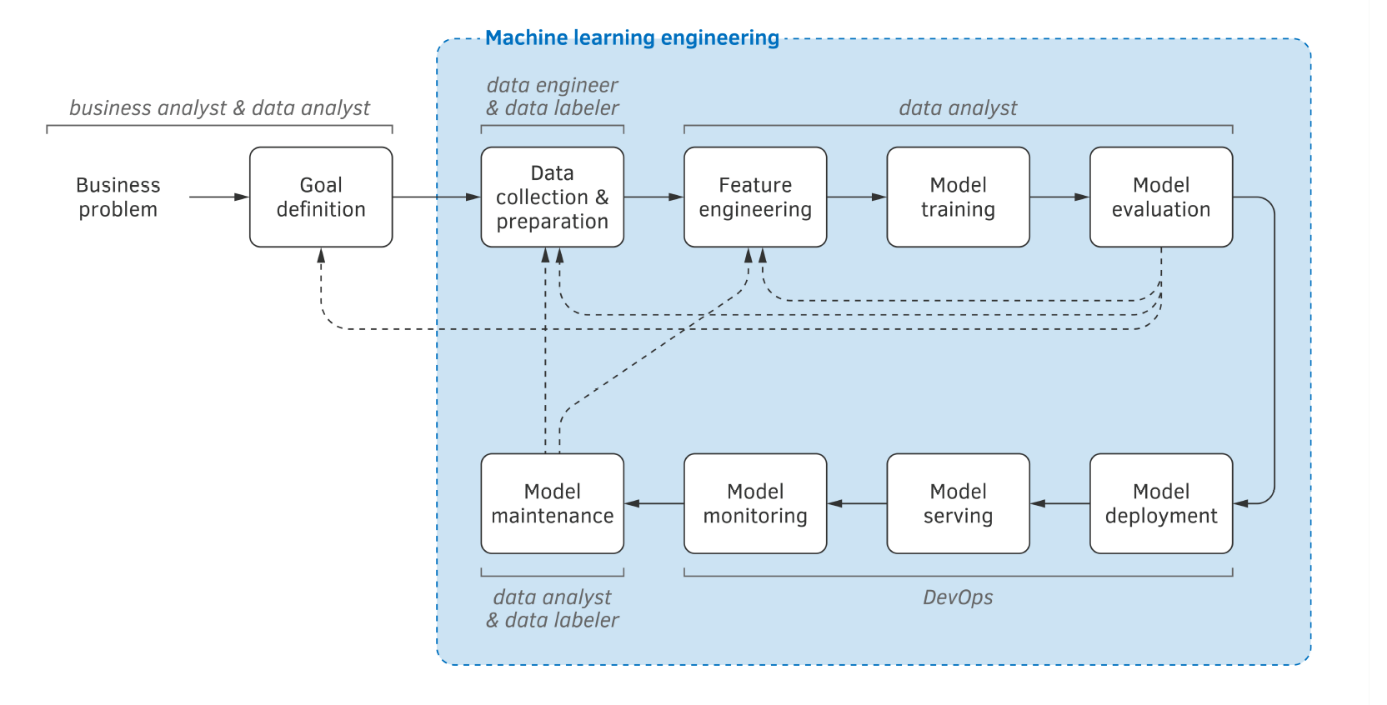


Figure 13: Machine learning life cycle

## Data collection & preparation

### Data collection

Dataiku DSS allows data scientists in FLOA Bank to deploy predictive models for real-time scoring using its Dataiku API node. The API node provides all the necessary features for scoring in production as input and allows users to query predictions as output.

Example of input of features data in real-time for Upcycle scoring:

*{ "clt\_address\_1": "string",*

*"clt\_address\_2": "string",*

*"clt\_address\_3": "string",*

*"clt\_address\_4": "string",*

*"clt\_birth\_date": "2019-10-25",*

*"clt\_birth\_zip\_code": "string",*

*"clt\_cell\_phone\_number": "string",*

*"clt\_city": "string",*

*"clt\_civility": "string",*

*"clt\_country": "string",*

*"clt\_customer\_ref": "string",*

*"clt\_email": "user@example.com",*

*"clt\_first\_name": "string",*

*"clt\_last\_name": "string",*

*"clt\_maiden\_name": "string",*

*"clt\_nationality": "string",*

*"clt\_phone\_number": "string",*

*"clt\_zip\_code": "string",*

*"nav\_ip\_address": "string",*

*"nav\_origin": "string",*

*"nav\_user\_type": "string",*

*"nav\_visit": "string",*

*"ppr\_decision\_bris\_cd1": "string",*

*"ppr\_decision\_bris\_cd2": "string",*

*"ppr\_decision\_bris\_cd3": "string",*

*"ppr\_decision\_bris\_cd4": "string",*

*"ppr\_decision\_cha\_cod\_reaf": "string",*

*"ppr\_decision\_cha\_hn1": "string",*

*"ppr\_decision\_cha\_hn2": "string",*

*"ppr\_decision\_cha\_hn3": "string",*

*"ppr\_decision\_cha\_hn4": "string",*

*"ppr\_decision\_cha\_not": "string",*

*"ppr\_decision\_sco\_note": 0,*

*"ppr\_journey\_contract": "string",*

*"ppr\_journey\_revenue": 0,*

*"ppr\_journey\_shelter": "string",*

*"request\_date": "2019-10-25T15:08:38.701000Z",*

*"request\_id": "string",*

*"request\_partner\_code": "string",*

*"request\_score\_type": "string",*

*"white\_list": "string" }*

Example of output data predictions in real-time for Upcycle scoring:

*{"\_\_meta\_\_":*

*{"api\_context":*

*{"endpointId": "traitement\_output\_cdp\_refus\_ppr",*

*"serviceGeneration": "RANDOM\_FOREST\_V10",*

*"serviceId": "SCORE\_RISQUE\_CDP\_REFUS\_PPR"},*

*"dss\_timing":*

*{"execution": 26350,*

*"functionInternal": 25651,*

*"preProcessing": 0,*

*"wait": 24}},*

*"decision": "R",*

*"input\_score": {*

*"request\_date": "2019-10-25T15:08:38.701000Z",*

*"request\_id": "string",*

*"request\_partner\_code": "string",*

*"request\_score\_type": "string",*

*"white\_list": "string"*

*},*

*"request\_date": "2020-10-25T15:08:38.701000Z",*

*"request\_id": "string",*

*"score\_called\_api\_context": {*

*"endpointId": "SCORE\_RISQUE\_CDP\_REFUS\_PPR",*

*"serviceGeneration": "RANDOM\_FOREST\_V10",*

*"serviceId": "SCORE\_RISQUE\_CDP\_REFUS\_PPR"},*

*"score\_called\_result":*

*{"ignored": false,*

*"prediction": "1",*

*"probaPercentile": 92,*

*"probas": {*

*"0": 0.6326911449614383,*

*"1": 0.36730885503856175}},*

*"score\_called\_timing":*

*{"enrich": 11,*

*"postProcessing": 30,*

*"preProcessing": 38,*

*"prediction": 1095,*

*"preparation": 19590,*

*"wait": 17 },*

*"score\_note": 2,*

*"score\_point": 0,*

*"score\_proba": 0.36730885503856175,*

*"seuil": 0.2624*

*}*

The model is built from features of the input data which contains the following types of information:

* Client information: Age, first name, last name, maiden name, city, address, zip code, email, country, nationality and phone number.
* Order information: origin of the demand, user type, visit type.
* Personal loan scoring information (PPR / PPC): decision score, revenue, shelter and contract.

In addition, open datasets can be used to enrich the input data to build predictive models. This process takes a column containing a French department code and outputs several columns with demographic data about this department such as:

* Basic demography data (population, households, births, deaths, …)
* Housing data (households, second homes, …)
* Fiscal/Revenue data (number of ‘foyers fiscaux’, average revenue, …)
* Employment data (number of jobs, unemployment rate, …)
* Companies data (number of companies, breakdown by industry sector, breakdown by size, …)

For Coup de Pouce product, we define potential clients based on several modalities determined as follows:

* Unknown: Clients signing up for a Coup de Pouce loan for the first time.
* Trusted prospect: Non-repeaters clients who had done at most only one coup de Pouce before without default or had been eligible for the loan but didn’t validate their order over the last 70 days.
* Trusted client: Repeaters clients who had done at least more than one Coup de Pouce before without default.
* Whitelist: VIPs and good clients.

In production, the score is used only for Unknown clients. Therefore, it is recommended to train a scoring model on at least 1.000 default cases of Unknown clients in order to build a robust model for future unknown clients.

In our case, our data set will cover the period starting from 2019/11/13 to 2021/05/23. The frequency distribution of Upcycle Coup de Pouce client ‘s modalities and default cases by modality is as follow:

Figure 14: Coup de Pouce UPCYCLE client’s modalities

Unfortunately, we have only 475 unknown clients with a default, representing 14% default rate in this client modality, which won’t be representative in our data set for the new model. As a result, we need to add another close modality to reach 1000 default cases: We add the trusted prospect clients.

To test whether the default rate of the two population (Unknown and Trusted prospect) is equal, we proceed by Student t-test at 90%, 95% and 99% confidence level:

|  |  |  |  |
| --- | --- | --- | --- |
| Samples | | | |
| *Client Modality* |  | *Count* | *Mean* |
| Unknown |  | 3384 | 14,04% |
| Trusted Prospect |  | 6039 | 14,19% |
|  | | | |
| Total |  | 9423 | 14,14% |

Figure 15: Proportion of Unknown vs Trusted prospect

|  |  |
| --- | --- |
| Hypothesis | |
| *Tested hypothesis* | *Default rate means are identical in both populations* |
| *Significance level* | 0.1 |
| *Significance level* | 0.05 |
| *Significance level* | 0.01 |
| Results | |
| *𝘵-statistic* | -0.206431921 |
| *𝘱-value* | 0.8364579992 |

At confidence level of 90%, 95% and 99%: We deduce non-significant difference in means among both populations. Therefore, our data set will cover Unknown and Trusted prospect clients, representing 9432 validated orders with 1332 (14.13%) default cases.

We donate our target variable “casse\_r2” a binary variable representing the default event as 1 in the case of default and 0 otherwise. The prediction type is “Two-class classification” and the proportion of the target classes is as follow:

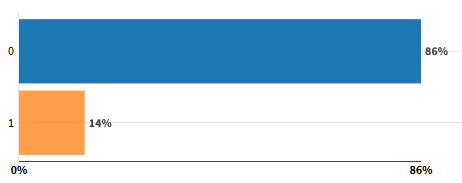
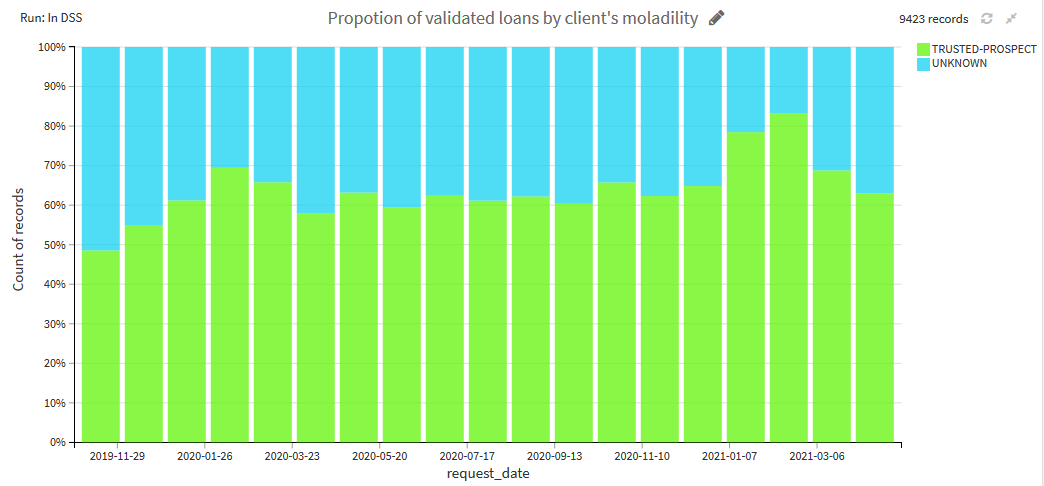


Figure 16: Proportion of default cases on the data set

### Data analysis

Starting with the proportion of validated orders and default cases by client’s modality. We notice a stable distribution of validated orders and default cases by modality, mostly from Trusted-Prospect clients, across the period selected for the modeling.



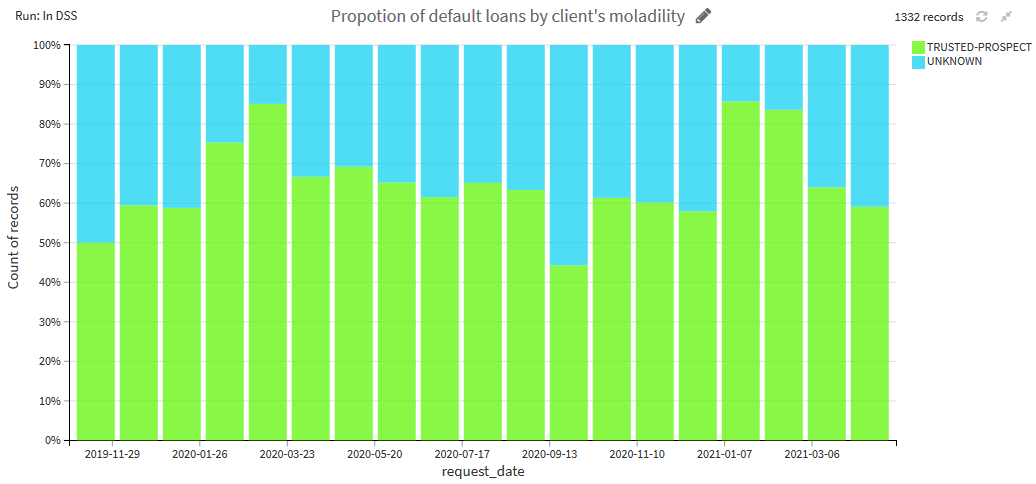


Figure 17: Proportion of validated and default loans by client’s modalities

The waterfall chart helps uderstanding the the gradual transition of Upcycle score path. From 448k request orders, only 13k are granted. The score refuted 215k request order where as 188K clients have abondenned their Coup de Pouce demand.

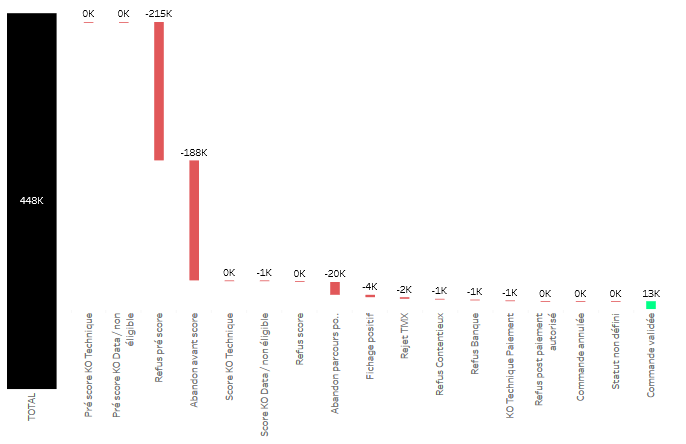
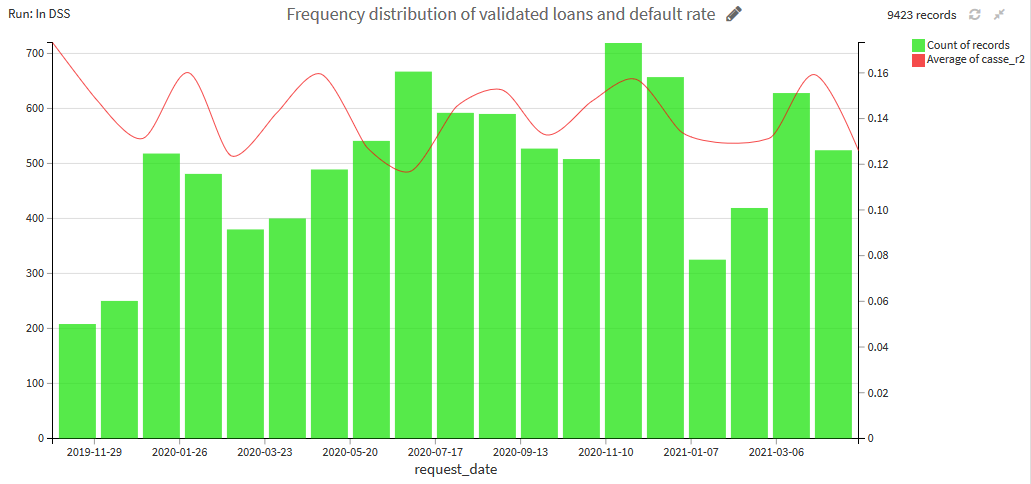


Figure 18: Waterfall chart of UPCYCLE scoring process

Among the validated orders, we can see that their distribution is stable, ranging from 400 to 700 commands per month. The average default rate remains high and also stable ranging between 13% and 16% default rate.

Figure 19: Granted Coup de Pouce loans by average default

The variable ‘ppr\_journey\_contract’ describes the client’s employment contracts: CDI, CDD, Intermittent contract... Most of Upcycle clients don’t fill in their employment contract status. A large proportion of clients are on CDI and the rest on CDD, INT or others employment forms.

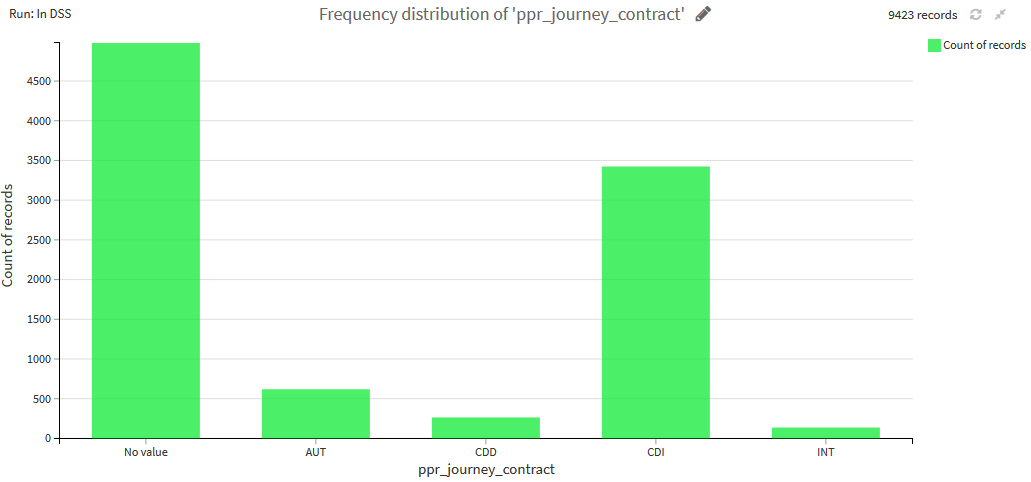


Figure 20: Client’s employment contracts

The variable ‘ppr\_journey\_shelter’ describes the housing occupancy status of the client: Locataire, propriétaire, logé par la famille, etc.. As seen below, most of Upcycle clients also don’t fill their housing status. A large proportion of clients are Locataire.

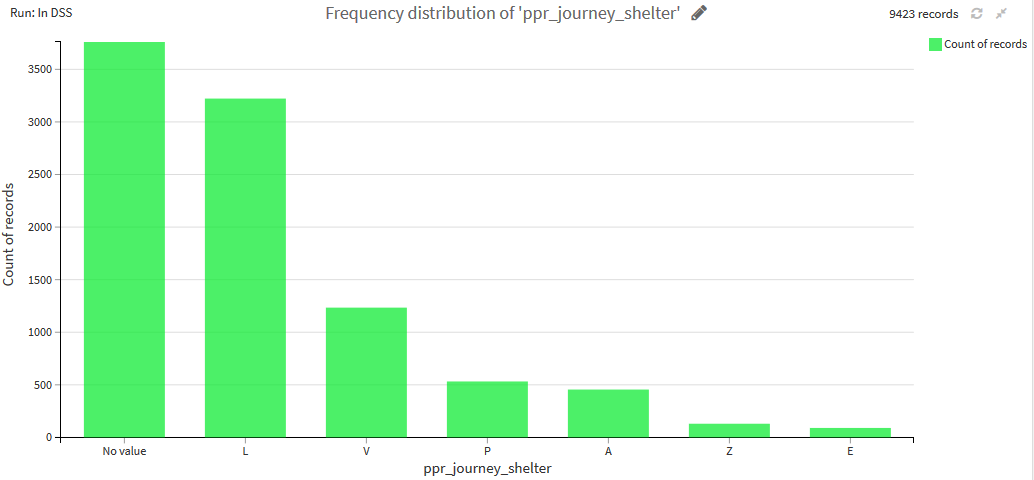


Figure 21: Client’s housing status

The variable ‘ppr\_decision\_sco\_note’ describes the score given by the credit score loan PPR of the original client demand. Unfortunately, this variable is not well mapped in our logs as it is usually missing.

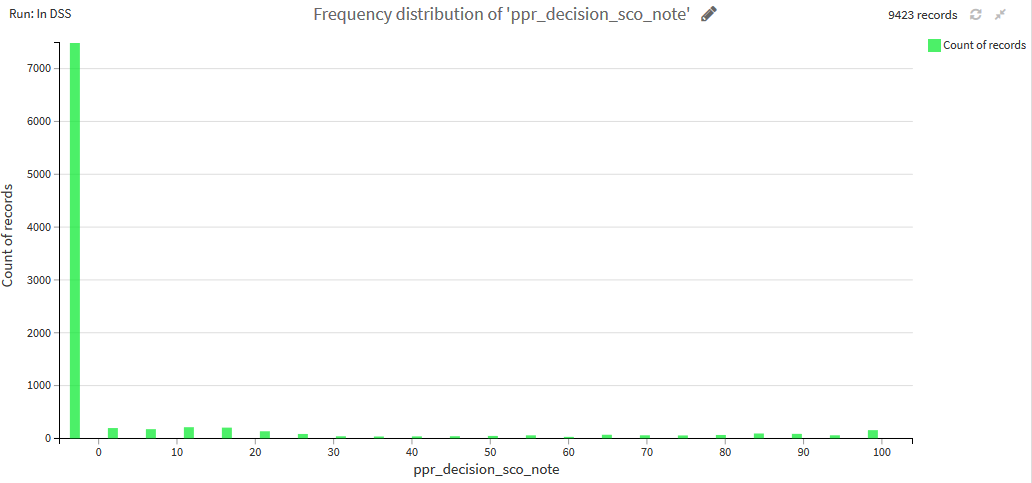


Figure 22: Client’s credit score on larger loans PPR/PPC

## Feature engineering

After data collection and preparation, feature engineering is the second most important activity in score modeling. We illustrate several techniques used for our scoring model:

### Binning and weights of evidence

The reasons to discretize a real-valued numerical feature can be numerous. A successful discretization adds useful information to learning algorithm when the training is relatively small and may increase the performance of the model (Szepannek,2007)

Binning of numeric variables is often considered to be the most relevant part of score modeling. In general, cluster analysis centers around reducing the Euclidean-distance of within cluster variation of observations, while maximizing the segregation among the different cluster groups/bins. However, the Euclidean distance is the “ordinary” distance between two points. To solve the problem, we propose to cluster the factor observations based on their Weight of Evidence

The Weight of Evidence (WoE) is widely used in credit scoring to separate good and bad clients. It compares the proportion of good accounts to bad accounts at each attribute level, and measure the strength of the attributes of an independent variable in separating good and bad accounts. As we cluster the factor attributes based on their corresponding WoE, the resulting bins would be analogous to default risk, regardless the fact that unsupervised learning methods don’t take any response variable into consideration.

Using the 'smbinning' package in R, this method uses Recursive Partitioning through conditional inference trees approach and can be summarized as follows:

1. Evaluate which variable is most significant by the following line code:

*filter(smbinning.sumiv(df = mydataset, y = "good"),Process=="Numeric binning OK")*

1. Evaluate the best split of a continuous variable through the Chi-square decision tree approach: When binning a continuous variable, we are predicting the target variable using only our one continuous variable in the conditional inference tree. It evaluates if the variable is significant at predicting the target variable. If so, it finds the most significant statistical split using Chi-square tests in between each value of the continuous variable and then comparing the two groups formed by this split. After finding the most significant split we have two continuous variables - one below the split and one above. The process repeats itself until the algorithm can no longer find significant splits leading to the definition of the bins.

* Count the number of default and non-default events for each attribute levels in the factor variable.
* Calculate WoE of each levels of attributes (WoE for the fine classes).
* Cluster the attributes based on their calculated WoE.
* Form the final bins according to the cluster groups.

The ‘smbinning’ function inside the smbinning package is the primary function to bin continuous variables and takes as the following arguments:

* *df* : Data frame
* *y* : Binary response variable (0,1). In our case, the default variable ‘casse\_r2’.
* *x* : Continuos characteristics
* *p* : Percentage of records per bin. We set the default 5%.

In our case study, the predictor variables defined by the function ‘smbinning’ set to be binned are:

* clt\_age: Client’s age.
* insee\_owner\_rate: Owner rate giving a postal code.
* insee\_dss\_ets\_indus\_2010: Number of industrial establishments giving a postal code in 2010
* insee\_dss\_ets\_construc\_2010 : Number of establishments in the construction sector giving a postal code in 2010
* clt\_email\_digits\_length: Length of digit numbers used in mail address.
* insee\_dss\_ets\_10plus\_201 : Number establishments with 10 or more employees giving a postal code in 2010
* insee\_pop1564: Total population between 15 and 64 years old giving a postal code.
* insee\_pop: Total population giving a postal code.
* insee\_chom1564: Number of unemployed giving a postal code.
* insee\_1564\_rate: Unemployment rate giving a postal code.

In our report, we choose the variable ‘clt\_age’ to illustrate this method. The ‘ivtable’ method in ‘smbinning’ package gives a summary of the splits as well as some information regarding each split. Working from left to right, the columns represent the number of observations in each bin, the number of goods (non-defaulters) and bads (defaulters) in each bin, as well as the cumulative versions of all of the above. Next comes the percentage of observations that are in the bin as well as percentage of observations in the bin that are both good and bad. Finally, the table lists the odds, natural log of the odds, weight of evidence (WoE), and information value component (which we are going to detail in following chapters).



From the chart below, we can deduce higher proportion of default cases among young clients under 27 years old where the default rate is above the average 14.14%. Whereas, the proportion of good loans are among clients older than 39 years old.

Moreover, WoE summarizes the separation between the event of default and non-default (bad and good loans) as shown in the graph below. For WoE we are looking for large differences in WoE between bins. Ideally, we would like to see monotonic increases for variables that have ordered bins. This isn’t always required as long as the WoE pattern in the bins makes business sense. However, if a variable’s bins go back and forth between positive and negative WoE values across bins, then the variable typically has trouble separating goods and bad loans.   
WoE approximately zero, client’s age between 39 and 48 years old, implies percentages of non-default (good loans) are approximately equal to percentages of default (bad loans) so that bin doesn’t separate well the default and non-default events. WoE of positives values, client’s age above 48 years old, implies the bin identifies observations that are non-default (good loan), while WoE of negative values, client’s age below 27 years old, implies bin identifies observations that are events (bad loans).

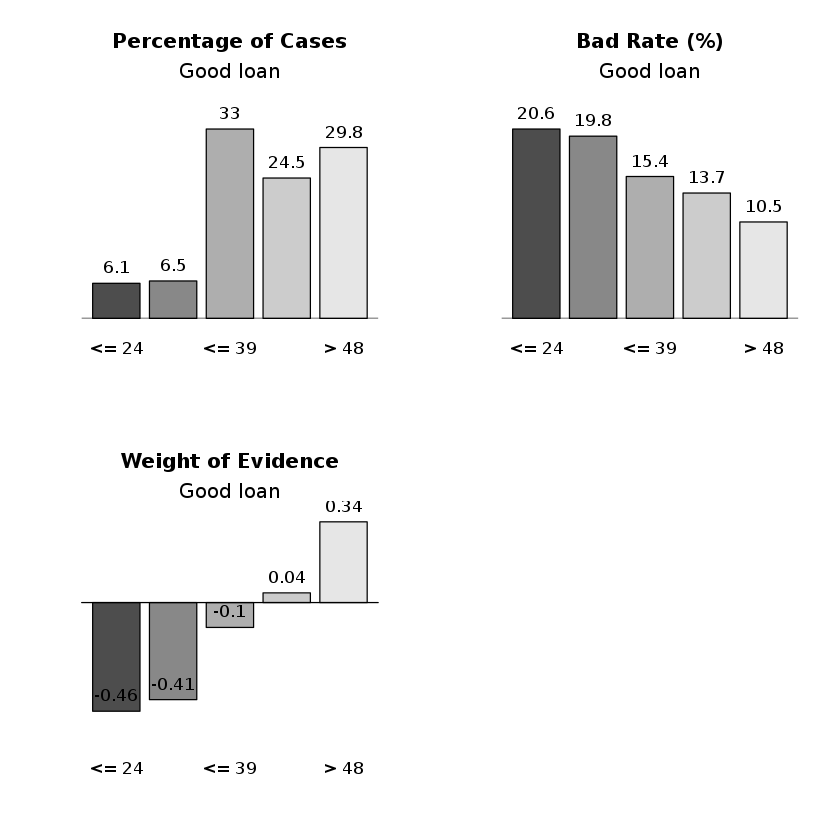
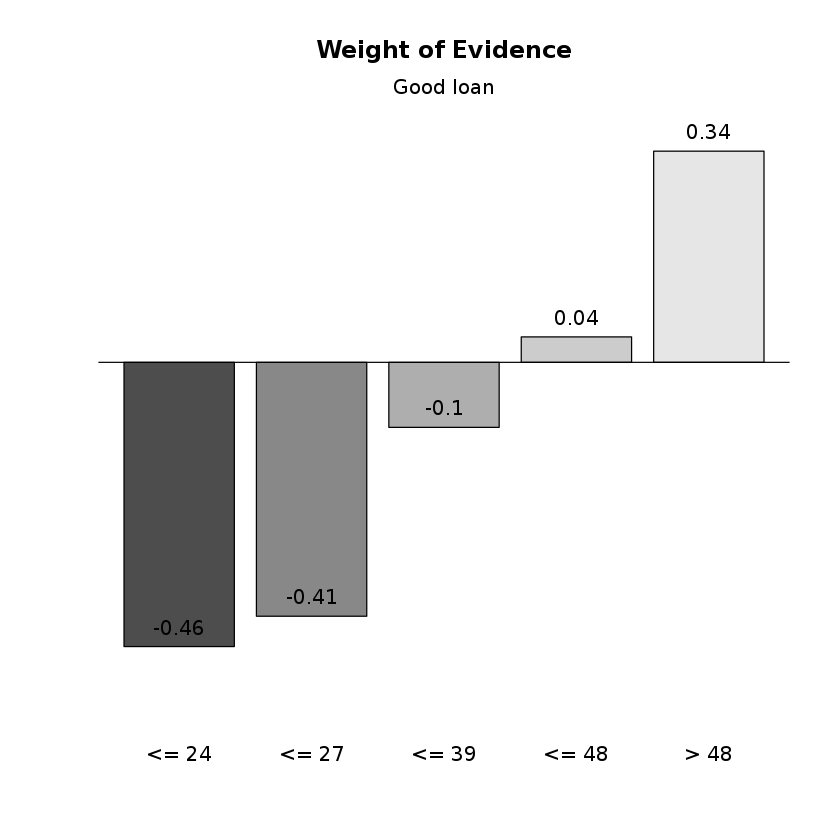


Figure 23: Smbining graphs

We then create a new categorical variable which contains age classes defined by the ‘smbinning’ and we call it ‘flg\_age\_risky2’: Class 1- Under 24, Class 2- between 24 and 27, Class 3- between 27 and 39, Class 4- between 39 and 48, Class 5- above 4\_ years old.  
  
In the other hand, based on the average rate of default cases in the ‘ivtable’ below, we can construct a new variable, called ‘flg\_age\_risky’ that takes two classes: Class 1- negative WoE and Class 2- positive WoE.



Class\_age\_2

Class\_age\_1

Figure 24: Weight of evidence cut off

Another approach is to use the WoE for encoding where we can replace the classes with their associated WoE values. However, this method had several drawbacks and results to overfitting as we manipulate the effect of variables according to how categories are created.

### Manuel binning

Manual binning offers detailed and tailored solution to classification within variables in line with business sense. For example, a variable can be binned according to the target group study by the firm, and such binning will make more business sense than some random classification based on equal size or equal weight binning. Especially for categorical variables, employing local expert knowledge has clear advantage over automated binning.   
We illustrate this method with the variable ‘nav\_origin’. This variable describes business providers of Coup de Pouce Upcycle and contains multiple modalities that are not stable across the study period:

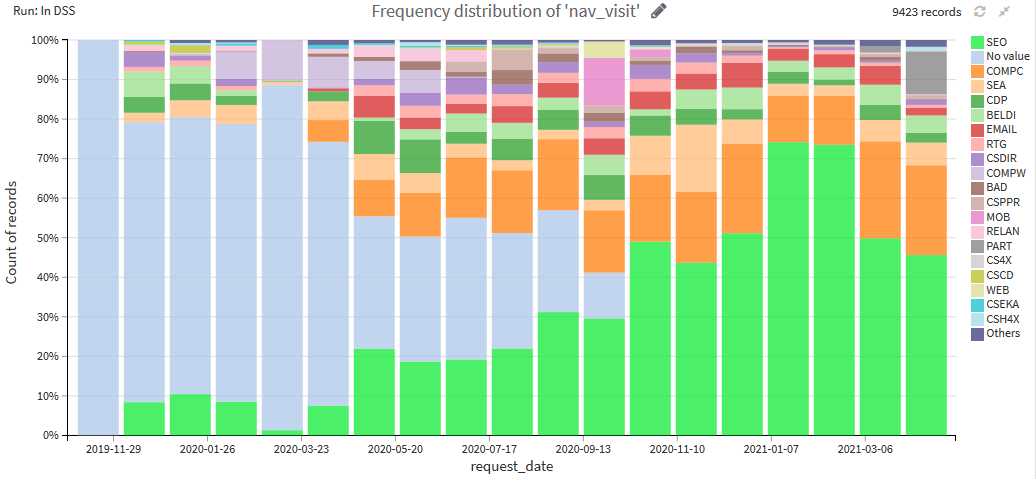


Figure 25: Navigator origin’s modalities of Coup de Pouce UPCYCLE demand

The marketing team has more knowledge on characteristics within this variable and a proposal solution was to define whether the demand is from FLOA client or a new potential client based on origin that stars with

* ‘CS’=cross sell, and ‘Bel’= banque en ligne : defines FLOA clients.
* ‘Comp’ : defines potentiel clients.
* ‘Email’ : could define either clients or potential clients.
* ‘CDP’ = coup de pouce et ‘Cdis’= cdiscount : could define either clients or potential clients.

Based on this, we can define a new variable ‘nav\_origin\_group’ that takes three modalities ‘Client’, ‘Potential client’ or ‘Potential client or Client’.

We proceed by Chi-square test to check the dependency between our new variable ‘nav\_origin\_group’ and the target variable ‘casse\_r2’ to evaluate the binning proposal:

|  |  |
| --- | --- |
| Hypothesis | |
| Tested hypothesis | nav\_origin\_group and casse\_r2 are independent |
| Significance level | 0.05 |
| Results | |
| Chi-square statistic | 7.3692719671 |
| Degrees of freedom | 2 |
| 𝘱-value | 0.0251063122 |

From this test, we conclude that variables ‘nav\_origin\_group’ and ‘casse\_r2’ are not independent. Hence this method results in precise binning of the variable by groups with similar characteristics.  
However, manual binning is a tedious and time-consuming process, especially when the number of predictor variables is large.

### Cluster Binning for Categorical Variables

Using Python, we compute the target proportion in each modality of qualitative features and assess the significance of the difference from the average target proportion. The proportion of each modality must be representative (>1% of total records) and the difference from the average using chi-square / ANOVA test has to be statistically significant at 5% confidence level.

We illustrate this method using email address variable ‘clt\_email\_domain\_0’. The output is as follow:



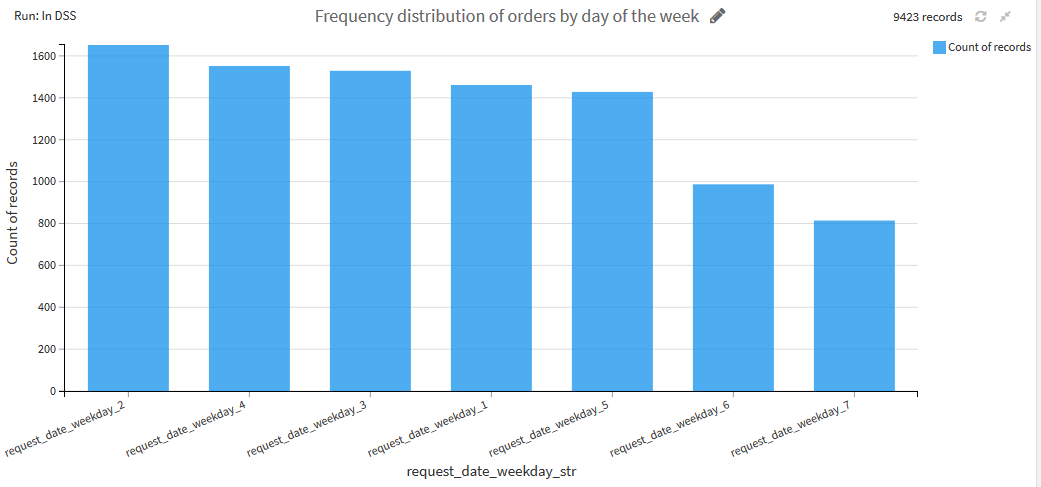
Based on that, we can define two dummy variables ‘clt\_email\_1’ which refers to less risky domain emails (default rate below the average 14% denoted by green dotes) and takes 1 if email is containing ‘waradoo, free, sfr, orange’ and ‘clt\_email\_2’ which refers to more risky domain emails (default rate above the average 14% denoted by red dotes) and takes 1 if email is containing ‘outlook, gmail’.

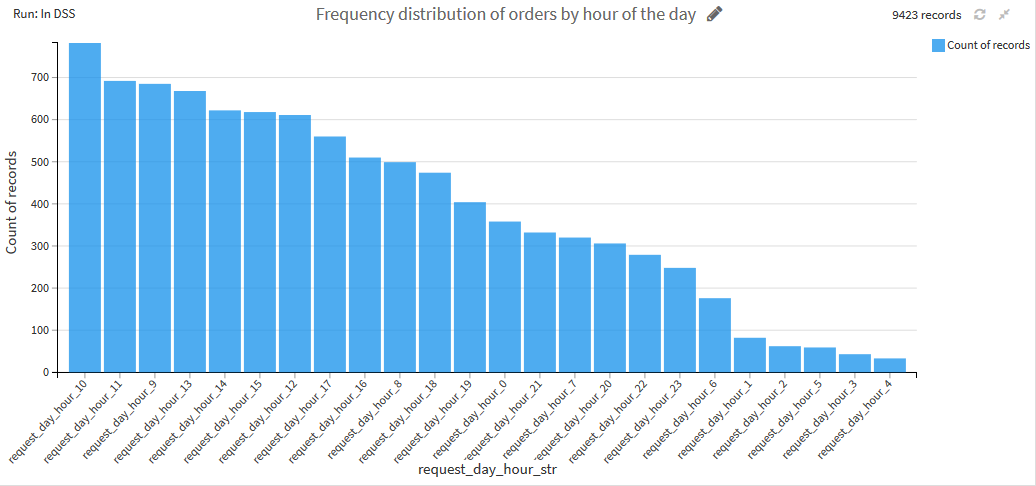
### Time Features

From the date of the order ‘request\_date’, we can extract valuable information such as:

* Order month: Ranging from 1 to 12.
* Order day of the year: From 1 to 356.
* Order day of the week: From 1 to 7.
* Order day of the month: From 1 to 30.
* Order hour: From 1 to 24.
* Order date zone.
* If the order was on school holiday or on bank holiday, etc..

The following charts illustrate the frequency distribution among the day of the week, hour of the day and the day of the month. We can see that most Coup de Pouce Upcycle demands are done mostly on mornings, during weekdays at the beginning of each month.





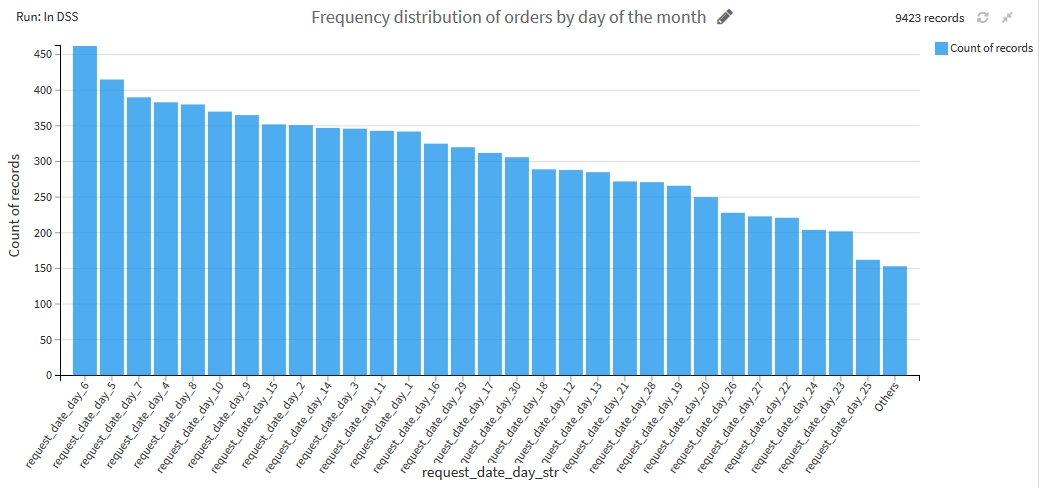


Figure 26: Distribution of order by time modalities

### Hamming distance

A distance between two strings of equal length is the number of positions at which the corresponding symbols are different. In other words, it measures the minimum number of *substitutions* required to change one string into the other, or the minimum number of *errors* that could have transformed one string into the other.

We calculate the distance between the username in the email and the first / last name of the client in order to detect similarity between them.

### Impact coding

When the number of categories becomes very large, dummy encoding may become inconvenient. An alternative method to clustering or truncation of categories consists in characterizing the categories by the link they have with the target variable y: this is impact encoding. This method is also known as likelihood encoding, target coding, conditional-probability encoding.

For a target variable y and categorical variable X with K modalities *,….*, each modality is encoded by its impact:

### Standard normalization

During the process of the feature engineering part, we used the standard normalization. It is the procedure during which the feature values are rescaled so that they have the properties of a standard normal distribution by subtracting the empirical mean and dividing by the standard deviation. It’s helpful to apply simple mathematical transformation to the feature values which includes taking the logarithm of the feature, squaring it, or extracting the square root of the feature. The idea is to obtain a distribution as close to a normal distribution as possible.

### Missing values

There are a few choices for handling missing values in categorical features:

* **Treat as a regular value** treats missing values as a distinct category.
* Replace missing values with the specified value and used for **randomly missing** data that are missing due to random noise by imputing the median value.

# Modeling

## Predictive power

As seen in previous chapters, Weight of Evidence summarizes the individual categories or bins of a variable. However, we need a measure of how well a variable do at separating the events from non-events. For that, we use the information value criteria (IV) as a measure of the predictive power of our variables.

In credit modeling, IV is used in some instances to actually select which variables belong in the model. Here are some typical IV ranges for determining the strength of a predictor variable at predicting the target variable:

* 0 ≤ IV <0.02 : Not predictor.
* 0.02≤ IV <0.1 : Weak predictor.
* 0.1≤ IV <0.25 : Moderate (medium) predictor.
* 0.25 ≤ IV : Strong predictor.

For categorical variables, IV uses the WOE from each category as a piece of its calculation:

To illustrate the predictive power of our categorical variables, we set an IV of 0.02 as a threshold as variables with information values less than 0.02 are typically not predictors in the scoring model.

Figure 27: Features by decreasing IV

As we can see, most of our variables are considered by the IV criteria as weak variables. The client department of residence and birth and client address are expected to have good predictive power in the model. In previous chapters, when defining the ‘smbinning’ package, we derive two variables from client’s age based on WoE: ‘clt\_age\_risky2’ taking the classes output of the function and ‘clt\_age\_risky’ where we define two classes based on the average default rate. From the chart above, we can see that classes defined by the ‘smbinning’ function have higher information value and thus more predictive power. As a result, we choose the following variable to integrate it in our model.

For numeric variables, we use Pearson correlation as a measure of their predictive power. We set a correlation threshold of 1.2%, in absolute value, with the target rate ‘casse\_r2’ and we set correlation threshold between the dependent variables to 80%.

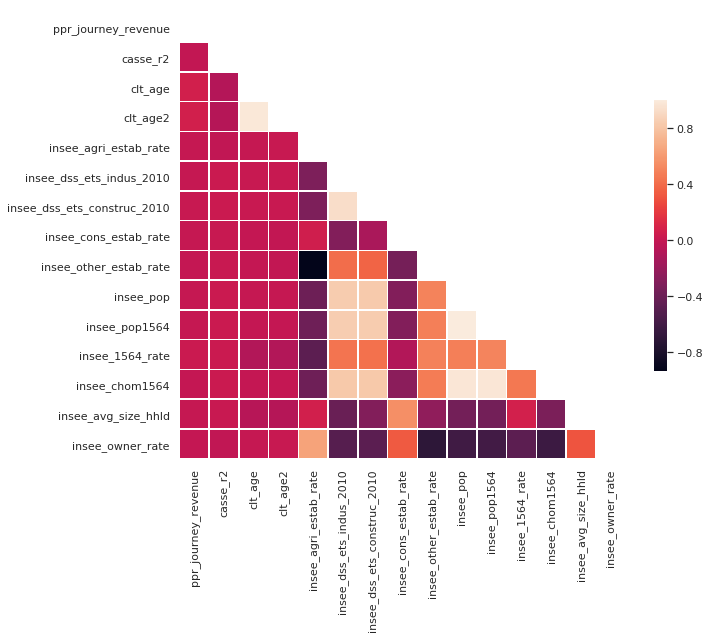


Figure 28: Pearson correlation matrix

## Performance metrics

The most common way to get a good model is to compare different models by calculation a performance metric on the holdout data. For classification, one way to assess the classification model performance is to use the “confusion matrix”, which summarizes how successful the classification model by comparing actual values (from the test dataset) to predicted values.

The confusion matrix is used to calculate the following performance metrics:

* + **Precision** is the ratio of true predictions to the overall number of positive predictions
  + **Recall** is the ratio of true predictions to the overall number of positive examples

In practice, we choose between high precision or high recall by varying the decision threshold for algorithms that return the prediction scores or by tuning the hyper parameters to maximize either precision or recall. This is called precision-recall tradeoff.

* + **F1-score** is the harmonic mean of precision and recall
  + **Accuracy** is given by the number of correctly classified examples, divided by the total number of classified examples. In terms of confusion matrix, it is given by:
  + **Hamming loss** is the fraction of labels that are incorrectly predicted

Where denotes the indicator function. Practically the smaller the value of hamming loss, the better the performance of the learning algorithm.

* + Roc curve: The Receiver Operating Characteristic (or ROC) curve shows the true positive rate TP/(TP+TN) versus the false positive rate FP/(FP+TN) resulting from different cutoffs in the predictive model. The “faster” the curve climbs, the better it is. On the contrary, a curve close to the diagonal line corresponds to a model with bad predictive power.

## Probabilities and lift

A binary classifier produces a probability that a given record is "positive" (here, that casse\_r2 is 1).  
The **lift** is the **ratio** between the results of the chosen model and the results obtained with a random model. Lift curves are particularly useful for "targeting" kinds of problems which will be the bad clients in our case. In FLOA Bank, the lift is one of the main criteria in the validation process of a model as we can focus on the region of scores where the cut-off is expected and, at a given level of rejection, we can measure how the scoring model is better than random selection. In practice, the calculation is done for Lift corresponding to 10%, 20%,,100% of clients with the worst score. For Upcycle score, we set a lift corresponding to 30% of bad clients.

The paper of Martin and František ŘEZÁČ – “How to Measure the Quality of Credit Scoring Models” helps expressing Lift by means of the cumulative distribution functions of the scores of bad and all clients. The empirical cumulative distribution functions (CDFs) of the scores of good (bad) clients are given by the relationships

where is the score of the client, n is the number of good clients, m is the number of bad clients, and is the indicator function, where (true) = 1 and (false) = 0. is the minimum value of a given score, is the maximum value.

The empirical distribution function of the scores of all clients is given by

The lift is defined as:

And the lift at a score level is defined as:

where q represents the score level of 100q% of the worst scores and can be computed as .

Since the expected rejection rate is usually between 5% and 20%, is typically assumed to be equal to 0.1 (10%), i.e., we are interested in the discriminatory power of a scoring model at the point of 10% of the worst scores. In our case, we set 30% lift, so we have

We illustrate the results by two charts:

* + - Cumulative Lift Curve chart

The goal of this curve is to visualize the benefits of using a model for targeting a subset of the population. On the horizontal axis, we show the percentage of the population which is targeted and on the vertical axis the percentage of found positive records.  
A **dotted diagonal** illustrates a random model (i.e., targeting 40% of the population will find 40% of the positive records).

A **wizard curve above** shows a perfect model (there are 15% positive records in our test set, so a perfect model would target only this)

The actual percentage of actual positives found by this model. The steeper the curve, the better.

* + - Per-bin lift chart

This chart sorts the observations by deciles of decreasing predicted probability. It shows the **lift** in each of the bins.

If there is 20% of positives in our test set, but 60% in the first bin of probability then the lift of this first bin is 3.

This means that targeting only the observations in this bin would yield 3 times as many positive results as a random sampling (equally sized bars at the level of the dotted line).

The bars should decrease progressively from left to right, and the higher the bars on first probability decile, the better.

## Hyperparameter tuning

Hyperparametrs is an important role on the model process. Some hyperparameters influence the speed of training. We use the Ramdom search and the metrics used to rank hyperparameter points are computed by cross-validation. In K-fold cross-validation the dataset is partitioned into k equally sized subsets. We select k=5 and then, 4 subsets are used as folded train sets while the remaining subset is retained to validate the model.   
This process is then repeated 5 times, once for each fold defined by the subset used as validation set.

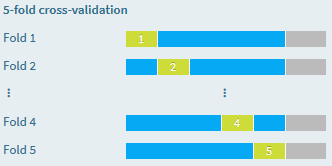


Figure 28: K-fold cross validation

## Machine learning algorithms

We partition our dataset into 70% of the data for the training and 30% for testing. The training set is used by the machine learning algorithm to train the model and the test set is used for reporting the upcoming performance results: During the training phase, DSS “holds out” on the test set, and the model is only trained on the train set. Once the model is trained, DSS evaluates its performance on the test set. This ensures that the evaluation is done on data that the model has “never seen before”.

We optimize our model hyperparameters for the cumulative lift and we optimize the threshold for the F1 score. We compute the lift at 30% and we define a cost matrix, which evaluates a ‘gain’ brought by this model based on the following hypothesis:

* + If the model predicts that the target variable ‘casse\_r2’ is true:
    - * And it is indeed true, the gain is 1
      * But it is not true, the gain is -0.3
  + If the model predicts that ‘casse\_r2’ is false:
    - * And it is indeed false the gain is 0
      * But it is actually true, the gain is 0

The algorithms selected for the training need to be in accordance with the following main properties:

* Explainability: The model require explanation for a non-technical audience. The most accurate machine learning algorithms and models are called ‘black boxes’. They make very few prediction errors but it may be difficult to understand and harder to explain. Examples of such models are deep neural networks or Ensemble models. In contrast, linear regression or decision tree learning algorithms are easy to interpret by a non-expert.
* Number of features and examples: Some algorithms can handle a huge number of examples and millions of features. Others can be relatively modest in their capacity.
* Training speed: Simple algorithms like random forest and logistic regression are robust and faster in the training. Others are slow to train and may need retraining frequently.
* Prediction speed: Selected models will be used in production where very high throughput is required when generating predictions.

### Random Forest

**Random Forest** is made of many decision trees. Each tree in the forest predicts a record, and each tree "votes" for the final answer of the forest. The forest chooses the class having the most votes. It’s a simple algorithm which builds a decision tree. Each node of the decision tree includes a condition on one of the input features. When "growing" (ie, training) the forest:

* for each tree, a random sample of the training set is used. We set number of trees between 100 and 700. The maximum depth of tree is between 1 and 3. And the minimum samples per leaf is set between 14 and 50.
* for each decision point in the tree, a random subset of the input features is considered. The algorithm default is set by selecting the square root of the number of features.

Random Forests generally provide good results, at the expense of explainability of the model.

### Gradient Boosted Trees

**Gradient boosting** is a technique which produces a prediction model in the form of an ensemble of "weak" prediction models (small decision trees). The concept is to train a set of decision trees (weak learners) to create a final strong learner. This is an iterative method. After each tree is trained, the data is reweighted: samples that were misclassified gain weight while the correctly classified ones lose weight. This allows future weak learners to focus on the "difficult" examples that the previous weak learners missed.

*Gradient Boosted Trees* is a generalization of boosting to arbitrary differentiable loss functions. GBT is an accurate and effective off-the-shelf procedure that can be used for both regression and classification problems. We set the number of boosting stages between 80 and 300. The feature sampling strategy is set to automatic mode by selecting 30% of the features and the learning rate is between 0.05 and 0.5. The advantages of GBRT are:

* Natural handling of data of mixed type (= heterogeneous features)
* Predictive power
* Robustness to outliers in output space (via robust deviance loss functions) and over-fitting

Due to the iterative nature of boosting, it is not very parallelizable and is less scalable than other algorithms.

### Logistic Regression

**Logistic Regression** is a classification algorithm using a linear model (i.e., it computes the target feature as a linear combination of input features). Logistic Regression minimizes a specific cost function (called *logit* or *sigmoid* function), which makes it appropriate for classification.   
A simple Logistic regression algorithm is prone to overfitting and sensitive to errors in the input dataset. To address these issues, it is possible to use a *penalty* (or *regularization term*) to the weights. This implementation can use either 'L1' or 'L2' regularization terms. We choose the L2 regularization with the corresponding inverse of the regularization parameter between 0.01 and 100.

## Experiment results

The obtained results are presented as follow:

### Confusion matrix and decision chart

* Random forest:



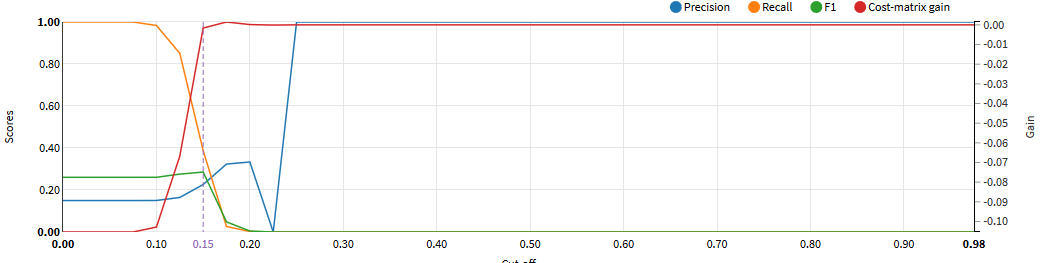


Figure 29: Confusion matrix and decision chart – Random forest

The optimal thresold to achieve the highest F1 score is set at 0.150 where we achieve 71% accuracy and 29% F1-score. Based on the model, the bank will lose customers who were denied a Coup de Pouce loan based on the model’s prediction that they would be defaulters.

* Gradiant boosting :



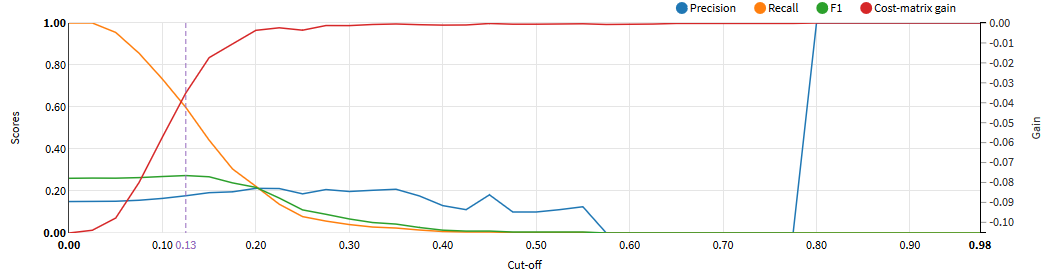


Figure 30: Confusion matrix and decision chart – Gradiant boosting

The optimal thresold to achieve the highest F1 score is set at 0.125 where we achieve 52% accuracy and 27% F1-score. The model correctly identifying True Positives at 60% rate. The precision is low at 18%.

* Logistic regression:





Figure 31: Confusion matrix and decision chart – Logistic regression

The optimal thresold to achieve the highest F1 score is set at 0.075 where we achieve 33% accuracy and 27% F1-score. The model correctly identifies 83% of default cases. The precision is low at 18%.

### Lift curve

* Random forest:



Figure 32: Cumulative Lift Curve chart – Random forest

This chart describes the ‘gain’ in targeting a given percentage of the total number of customers using the highest modelled probabilities of default, rather than targeting them at random.   
This model suggests that the top 30% (decile 1, 2 and 3) of customers with the highest predicted probabilities of default contain approximately 45% of non-payers rather than capturing 30% of non-payers.

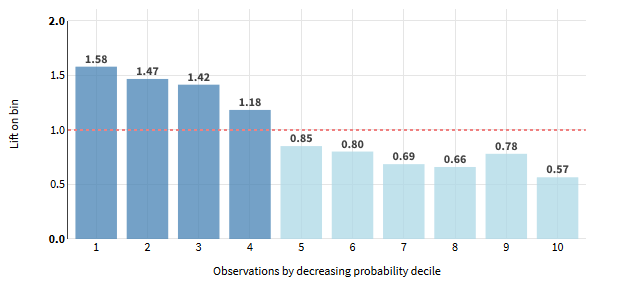


Figure 33: Per-bin lift chart – Random forest

The lift curve of this model is extended highly into the first decile. At decile group 1, 2 and 3, we have a lift of 1.49 which corresponds with there being 1.49 the number of non-payers captured compared with the number we would expect at random.

* Gradient boosting:

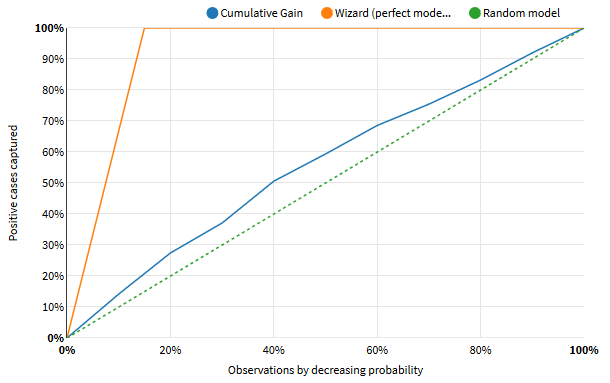


Figure 34: Cumulative Lift Curve chart – Gradient boosting

This model suggests that the top 30% (decile 1, 2 and 3) of customers with the highest predicted probabilities of default contain approximately 35% of non-payers rather than capturing 30% of non-payers.



Figure 35: Per-bin lift chart – Gradient boosting

On the other hand, targeting 30% of population by decreasing probability would results at identifying only 1.238 of the number of non-payers compared by a random model. The lift curve is not decreasing by decile as a good classifier has to give us a high lift when we act on only a few cases and as we include more cases the lift has to be decreasing.   
Also, the lift curve for the best possible classifier has to overlap the existing curve at the start which is not the case at the third decile where the lift is at 0.97.

* Logistic regression:

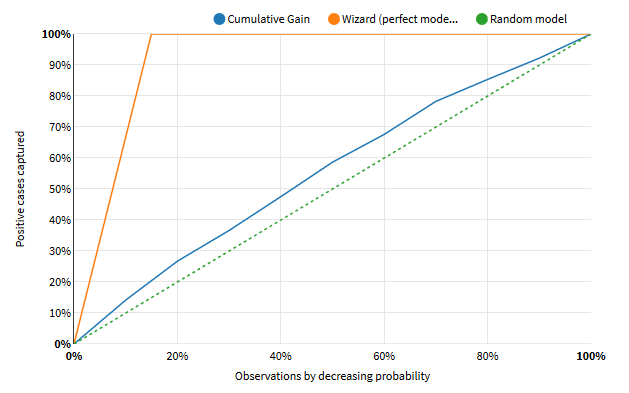


Figure 36: Cumulative Lift Curve chart – Logistic regression

This model suggests that the top 30% (decile 1, 2 and 3) of customers with the highest predicted probabilities of default contain approximately also 35% of non-payers rather than capturing 30% of non-payers.

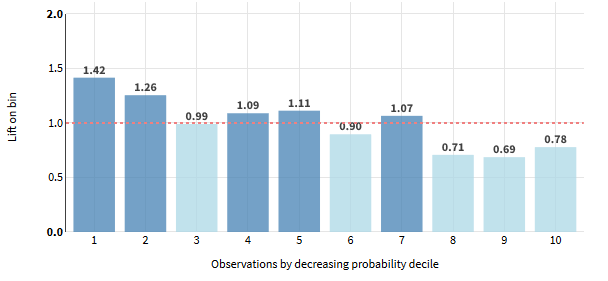


Figure 37: Per-bin lift chart – Logistic regression

On the other hand, targeting 30% of population by decreasing probability would results at identifying only 1.222 of the number of non-payers compared to a random model. Same as before, the lift curve is not decreasing by decile and overlay at the third decile with a lift of 0.99.

### ROC Curve

* Random forest:

The AUC (Area Under the Curve) for this model is 0.612, which is **fair**.

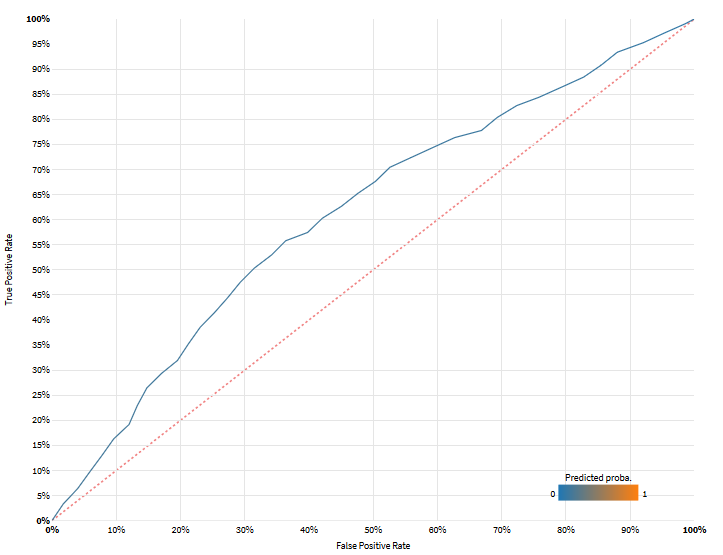


Figure 38: ROC Curve – Random forest

* Gradient boosting trees:

The AUC (Area Under the Curve) for this model is 0.571, which is **not very good.**

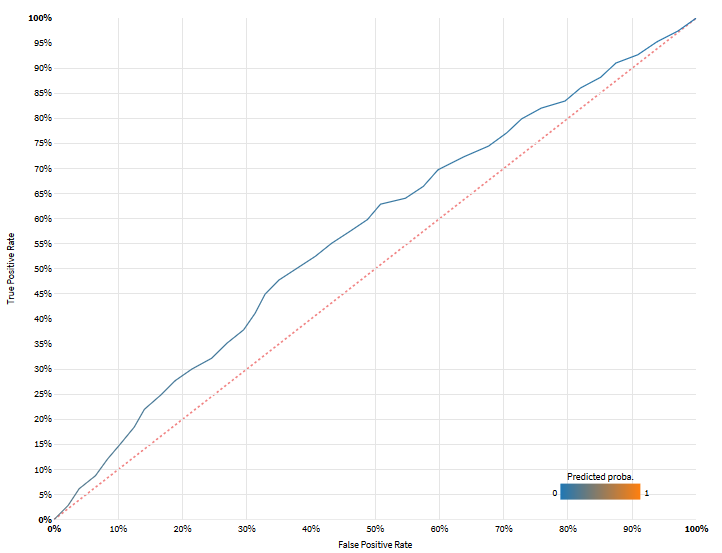


Figure 39: ROC Curve – Gradient boosting

* Logistic regression:

The AUC (Area Under the Curve) for this model is 0.570, which is **not very good.**

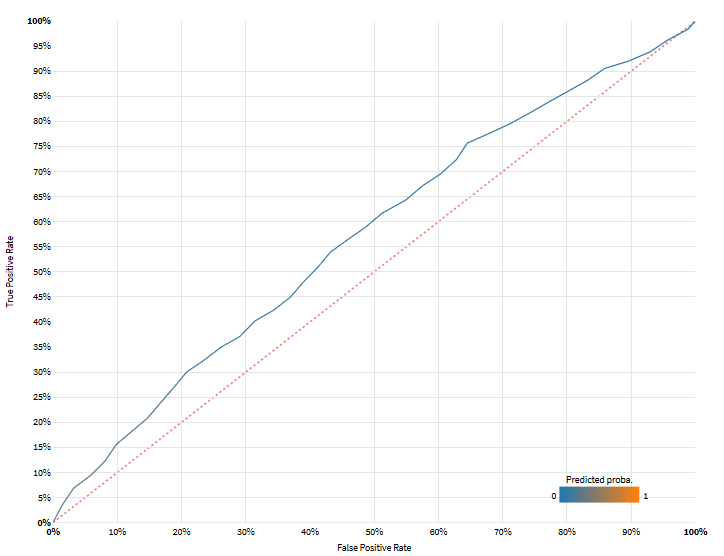


Figure 40: ROC Curve – Logistic regression

### Detailed metrics

* Random forest:

|  |  |
| --- | --- |
| Threshold-dependent (current threshold = 0.1500) | |
| **Accuracy** Proportion of correct predictions (positive and negative) in the test set | **0.7094** |
| **Precision** Proportion of positive predictions that were indeed positive (in the test set) | **0.2259** |
| **Recall** Proportion of actual positive values found by the classifier | **0.3877** |
| **F1 Score** Harmonic mean between Precision and Recall | **0.2855** |
| **Hamming loss** Fraction of labels that are incorrectly predicted (the lower the better) | **0.2906** |
| **Cost matrix gain** Average gain per record that the test set (2825 rows) would yield given the specified gain for each outcome. Specified gains: TP = 1, TN = 0, FP = -0.3, FN = 0. | **-0.0016** |
| **Matthews Correlation Coefficient** Correlation coefficient between actual and predicted values. +1 = perfect, 0 = no correlation, -1 = perfect anti-correlation | **0.1255** |
| Threshold-independent | |
| **Log loss** Error metric that takes into account the predicted probabilities (the lower the better) | **0.4175** |
| **ROC - AUC Score** Area under the ROC; from 0.5 (random model) to 1 (perfect model) | **0.6118** |
| **Calibration loss** Average distance between calibration curve and diagonal.  From 0 (perfectly calibrated) up to 0.5. | **0.0118** |

* Gradient boosting trees:

|  |  |
| --- | --- |
| Threshold-dependent (current threshold = 0.1250) | |
| **Accuracy** Proportion of correct predictions (positive and negative) in the test set | **0.5242** |
| **Precision** Proportion of positive predictions that were indeed positive (in the test set) | **0.1773** |
| **Recall** Proportion of actual positive values found by the classifier | **0.5981** |
| **F1 Score** Harmonic mean between Precision and Recall | **0.2735** |
| **Hamming loss** Fraction of labels that are incorrectly predicted (the lower the better) | **0.4758** |
| **Cost matrix gain** Average gain per record that the test set (2825 rows) would yield given the specified gain for each outcome. Specified gains: TP = 1, TN = 0, FP = -0.3, FN = 0. | **-0.0351** |
| **Matthews Correlation Coefficient** Correlation coefficient between actual and predicted values. +1 = perfect, 0 = no correlation, -1 = perfect anti-correlation | **0.0780** |
| Threshold-independent | |
| **Log loss** Error metric that takes into account the predicted probabilities (the lower the better) | **0.4292** |
| **ROC - AUC Score** Area under the ROC; from 0.5 (random model) to 1 (perfect model) | **0.5706** |
| **Calibration loss** Average distance between calibration curve and diagonal.  From 0 (perfectly calibrated) up to 0.5. | **0.0285** |

* Logistic regression:

|  |  |
| --- | --- |
| Threshold-dependent (current threshold = 0.0750) | |
| **Accuracy** Proportion of correct predictions (positive and negative) in the test set | **0.3292** |
| **Precision** Proportion of positive predictions that were indeed positive (in the test set) | **0.1621** |
| **Recall** Proportion of actual positive values found by the classifier | **0.8345** |
| **F1 Score** Harmonic mean between Precision and Recall | **0.2714** |
| **Hamming loss** Fraction of labels that are incorrectly predicted (the lower the better) | **0.6708** |
| **Cost matrix gain** Average gain per record that the test set (2825 rows) would yield given the specified gain for each outcome. Specified gains: TP = 1, TN = 0, FP = -0.3, FN = 0. | **-0.0688** |
| **Matthews Correlation Coefficient** Correlation coefficient between actual and predicted values. +1 = perfect, 0 = no correlation, -1 = perfect anti-correlation | **0.0635** |
| Threshold-independent | |
| **Log loss** Error metric that takes into account the predicted probabilities (the lower the better) | **0.4331** |
| **ROC - AUC Score** Area under the ROC; from 0.5 (random model) to 1 (perfect model) | **0.5703** |
| **Calibration loss** Average distance between calibration curve and diagonal.  From 0 (perfectly calibrated) up to 0.5. | **0.0411** |

Based on this analysis, the best model is Random forest which the highest 30% lift of 1.49: We illustrate the performance metrics on the Train and Test set:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *TEST\_precision* | *TEST\_recall* | *TEST\_auc* | *TEST\_f1* | *TEST\_accuracy* | *TEST\_lift* |
| 0.2258 | 0.3877 | 0.6118 | 0.2854 | 0.7093 | 1.490 |
| *TRAIN\_precision* | *TRAIN\_recall* | *TRAIN\_auc* | *TRAIN\_f1* | *TRAIN\_accuracy* | *TRAIN\_lift* |
| 0.2619 | 0.4884 | 0.6931 | 0.3410 | 0.7399 | 1.7788 |

We can identify if the model has overfit by first evaluating the model on the training dataset and then evaluating the same model on a holdout test dataset. If the performance of the model on the training dataset is significantly better than the performance on the test dataset, by computing the difference between the metrics obtained on Train and Test set, then the model may have overfit the training dataset. By looking at the difference ratio on AUC and Lift, we conclude that we model have a good performance and doesn’t overfit.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Diff\_Lift* | *Diff\_AUC* | *Diff\_precision* | *Diff\_recal* | *Diff\_accuracy* |
| 0.28 | 0.08 | 0.03 | 0.10 | 0.03 |

# Interpretation

## Feature importance

Several measures are available for feature importance in Random Forests:

**Gini Importance** or **Mean Decrease in Impurity (MDI)** calculates each feature importance as the sum over the number of splits (across all tress) that include the feature, proportionally to the number of samples it splits. Dataiku DSS extract feature importance provided by the fitted attribute feature\_importances\_ of Scikit-Learn and outputs the following result:

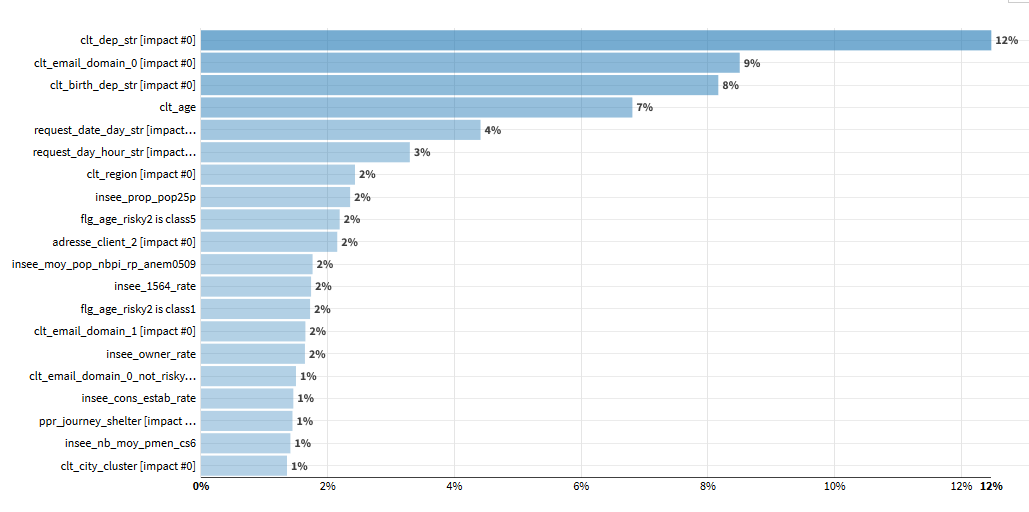


Figure 41: Feature importance

## Partial dependence

A partial dependence plot shows the dependence of the predicted response on a single feature. The x axis displays the value of the selected feature, while the y axis displays the partial dependence.   
The value of the partial dependence is by how much the log-odds are higher or lower than those of the average probability. In other words, the partial dependence value shows how the probability of being in the default class changes across different values of the feature.

The log-odds for a probability *p* are defined as *log(p / (1 - p)).* They are strictly increasing, ie. higher log odds means higher probability of default. We illustrate this method with several variables:

* + - clt\_depart\_str

We represent 30 most frequent modalities of ‘clt\_dep\_str’, computed on 2825 rows (the full test set)

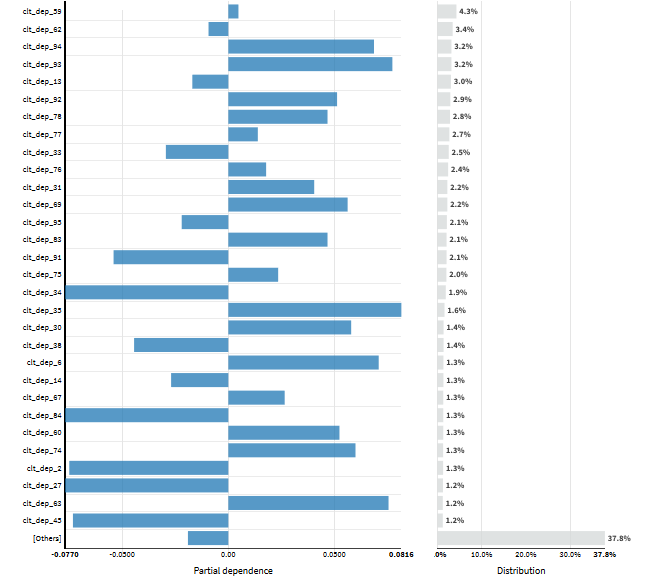


Figure 42: Partial dependency of client’s department

The client department is considered to be the most important feature by the model. We notice that several departments increase the probability of default whereas other departments are less likeliy to do so. The department of Val-de-Marne (94) representing 3.2% of the data increases the probability of default. In the other hand, the department of Gironde (33) representing 2.5% of the data decreases the probability of default.

* + - Insee\_prop\_pop25

We represent 50 bins for ‘insee\_prop\_pop25p’, computed on 2825 rows (the full test set)

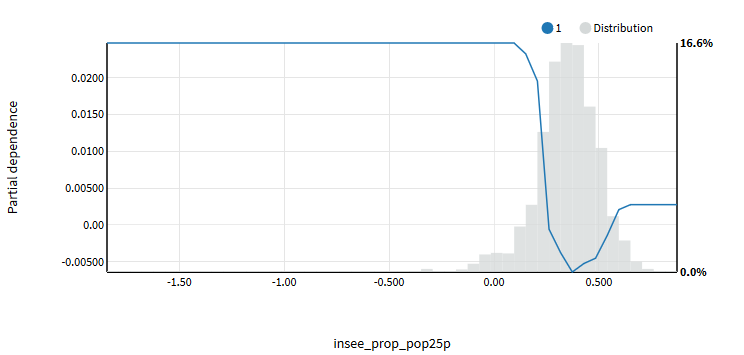


Figure 43: Partial dependency of the proportion of Insee population under 25 years old

The proportion of population under 25 years old ‘insee\_prop\_pop25’ is considered as an important feature in the model. Let’s investigate the following question: “To what extent does a client’s chances of defaulting depend on the proportion of the proportion of population under 25 years?”

Let’s first examine the trend line. We can see that as the proportion rate increases, the clients are steadily less likely to default. In addition to the trend line, we also have the distribution of the variable. We notice that the proportion of population under 25 years between 30% and 40% represents a major part of the data and we might be surprised that the probability of default goes up as this proportion exceed 50% but the small amount of data in this bracket could be a reason for the small dip.

* + - request\_date\_day\_str

We represent 30 most frequent modalities of ‘request\_date\_day\_str’, computed on 2825 rows (the full test set)

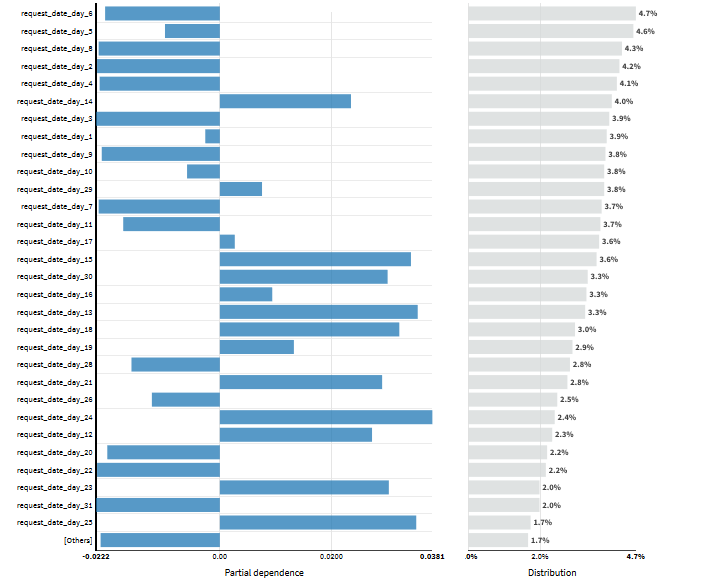


Figure 44: Partial dependency of the request day

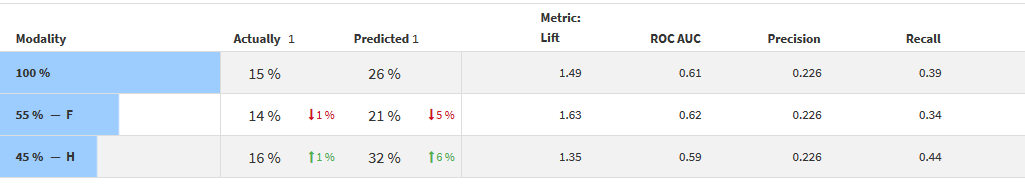
The day of the request is also considered to be an important feature by the model. We notice that orders done on first days of the month representing 16% of the data decreases the likelihood of default. On the other hand, orders done on the second part of the month, between 15 to 30, representing 15% of the data increases the likelihood of default.

## Subpopulation analysis

In this part, we investigate whether our model behaves identically across different subpopulation using different metrics like the lift, AUC, precision and the recall. We illustrate this method with the following variables:

* + - clt\_sex:

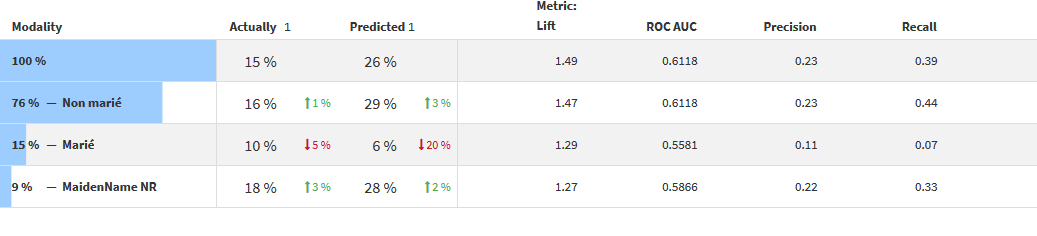
We have two modalities for ‘clt\_sex’ (Femme,Homme), computed on **2825** rows (test set).



For the two largest subgroups in this case, common metrics like lift, ROC AUC, accuracy, precision, and recall appear to be quite close. The female modality represents 55% of the data with 14% of default cases, 1% below the average default rate, among this modality. The model predicts 26% of unpaid loans among females which is 5% below the average predicated default rate. On the other hand, the male modality represents 45% of the data with 16% of default cases, 1% above the average default rate, among this modality. However, the model predicts 32% of unpaid loans among males which is 6% above the average predicated default rate.

* + - clt\_marie:

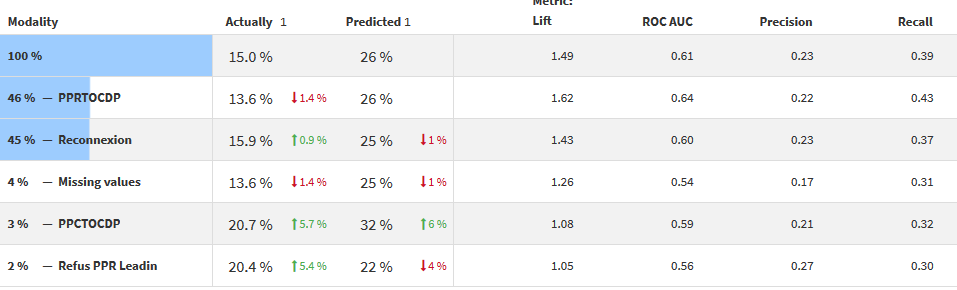
We have three modalities for ‘clt\_marie’ (Non marié, Marié, and Non renseigné), computed on **2825** rows (test set).



As seen before, common metrics like lift, ROC AUC, accuracy, precision, and recall appear to be quite close. The model predicts more default cases for non-married clients whereas for married clients seem less likely to default (only 6% of predict default cases among this modality).

* + - tracabilite:

This variable describes the origin of the demand, detailed in previous chapters, computed on **2825** rows (test set).



We notice that the model struggles to define the default from demands coming from ‘Refus PPR Leadin’ or where the origin is unknown. However, ‘PPRTOCDP’, which represents the large proportion of Coup de Pouce Upcucle demands, the model seems to better define the default event.

## Individual analysis

Individual prediction explanations are feature importance specific to a given sample. When the model is linear (logistic regression, OLS...), the explanation for one feature is simply the impact of the feature on the prediction with the mean feature value as a baseline: coefficient \* (feature value - mean feature value). As a generalization, the explanation is the difference between the prediction value and the average of prediction values obtained by replacing the feature value by values drawn from the test dataset. This method approximates Shapley values, trading off speed against both bias and variance.

For classification problems, the explanations are computed probability log-odd ratios: *log(p / (1 - p)).*

Using DSS Dataiku, we can compute the explanations for the most influential features for extreme probabilities using the method: Shapley Additive exPlanations. The goal of SHAP is to explain the prediction of an instance x by computing the contribution of each feature to the prediction. The SHAP explanation method computes Shapley values from coalitional game theory.

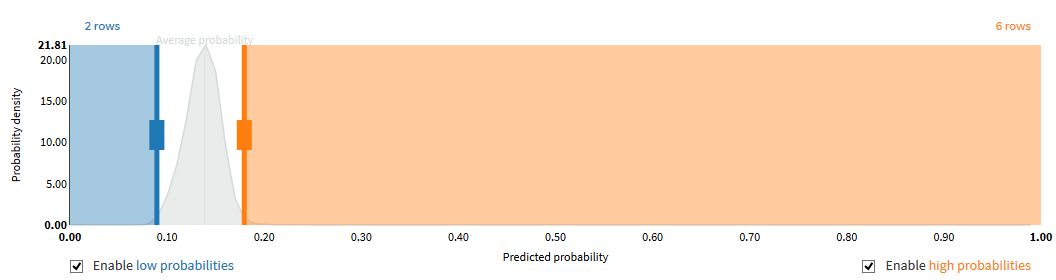
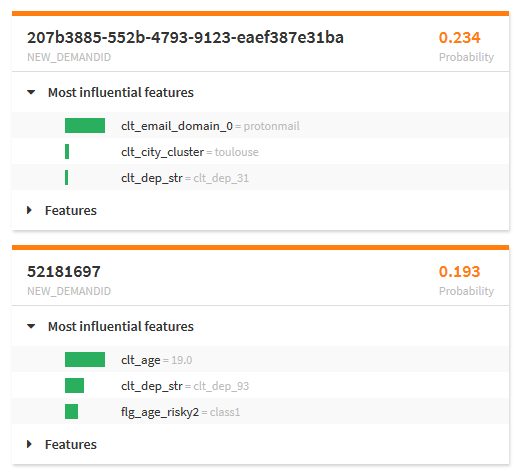
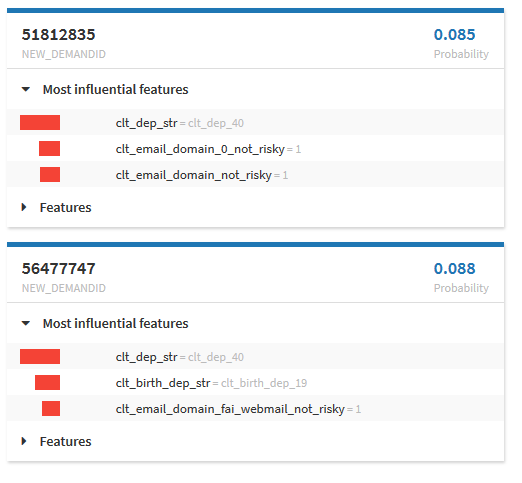


Figure 45: Probability density and predicted probability

We illustrate 4 examples with the most influential features of below the average of default probability (non-payment loan event) and above the average of default probability (payment loan event)



# Conclusion

In conclusion, one limitation of a credit scoring model is that the model output is only as good as the input that is used. If data going into the scoring model is inaccurate (for instance, client’s score on PPR and PPC loans or the origin of Coup de Pouce UPCYCLE demand, etc..), the model’s score output will be erroneous. Depending on how the erroneous information is weighted in the scoring formula, the impact on the score could be substantial.

The effectiveness of the model score output can also be constrained by factors such as changing economic conditions and business environments. As illustrated in previous chapters, business providers, managed by the commercial team, can vary frequently and bring different client’s profiles.

A model is typically developed for a certain target population and may be difficult to adapt to other populations. In most cases, a credit scoring model should only be used for the product, range of loan size, and market that it was developed for. Granting small loans like 500€ or 1000€ to refusals whose original loan’s demand is ranging between 5.000€ and 15.000€ can be challenging. When a bank tries to adapt the model to a different population, performance of that population may likely deviate from expectation.

Moreover, models are calibrated using historical data (in our case, we had to add a different modality, which is not going to be evaluated by the model in the production environment, in our credit score modeling). So, if relevant un-modeled conditions change, the model can have trouble forecasting out of sample.

# Appendix

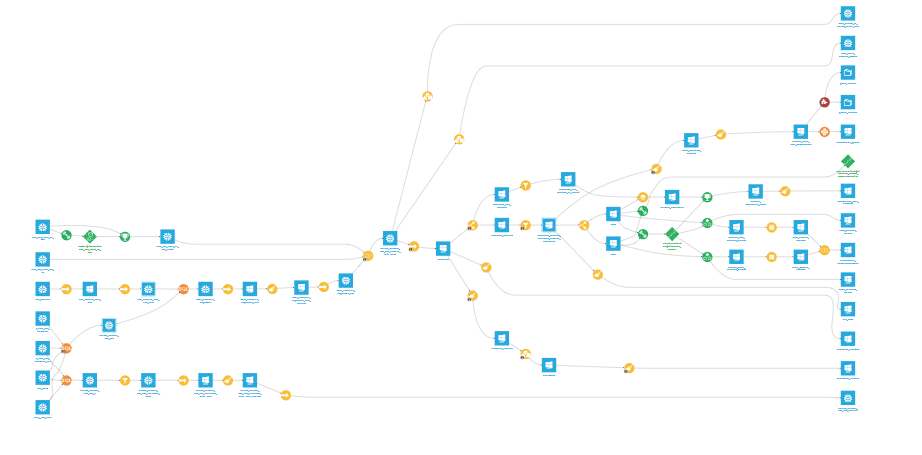


Figure 46: Dataiku flux of UPCYCLE score modeling

# Bibliography

<https://databricks.com/blog/2019/09/18/productionizing-machine-learning-from-deployment-to-drift-detection.html>

G. Szepannek. On the practical relevance of modern machine learning algorithms for credit scoringapplications.WIAS Report Series, 29:88–96, 2017.

<https://doi.org/10.20347/wias.report.29>

<https://journal.fsv.cuni.cz/storage/1228_rezac.pdf>

<https://www.mwsug.org/proceedings/2013/AA/MWSUG-2013-AA14.pdf>

<https://sundarstyles89.medium.com/weight-of-evidence-and-information-value-using-python-6f05072e83eb>

<https://www2.deloitte.com/content/dam/Deloitte/global/Documents/Financial-Services/gx-be-aers-fsi-credit-scoring.pdf>

<https://medium.com/crim/manipuler-les-variables-cat%C3%A9goriques-dans-un-jeu-de-donn%C3%A9es-6973c54c9827>

<https://cran.r-project.org/web/packages/smbinning/smbinning.pdf>

<https://www.ariclabarr.com/credit-modeling.html>

<http://papersjds14.sfds.asso.fr/submission_48.pdf>

<https://www.fdic.gov/regulations/examinations/credit_card/pdf_version/ch8.pdf>

<https://arxiv.org/pdf/2006.11835.pdf>

<https://www.researchgate.net/publication/309727706_Application_Scorecard_Modeling_Techniques_and_Performance>

<https://scikit-learn.org/stable/auto_examples/ensemble/plot_forest_importances.html>

<http://eric.univ-lyon2.fr/~ricco/tanagra/fichiers/fr_Tanagra_R_Python_PDP.pdf>

<https://www.aquiladata.fr/insights/shap-mieux-comprendre-linterpretation-de-modeles/>

<https://christophm.github.io/interpretable-ml-book/shap.html>