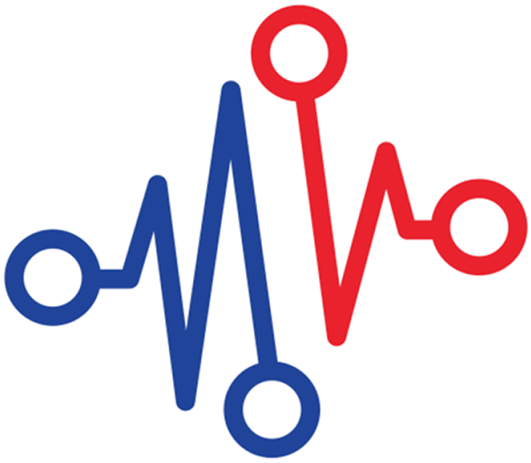
Report

Thomas Meunier

August 2021

Automatic or semi-automatic detection of companies in difficulty or weakened by the crisis



April 2021 - August 2021

Under the supervision of SignauxFaibles from the Ministère de l'Économie et des Finances

Contents

Abstract

In this report, we will attempt to improve a failure prediction model that is currently used by the French Ministry of Economy and Finance.

First, we studied several models and benchmarked them in order to compare our results with those of the articles studied. As a result, we were able to select four models that stood out from the rest, and that we should improve as much as possible.

Secondly, we decided to look at the data itself. We realized that our dataset was static. That is, for each row in our table, we had data only for a time T. We therefore decided to add variables, which we will call temporal features, in order to take temporality into account in the model. This addition was more than conclusive, because it allowed us to obtain excellent results that had not been achieved until then. Afterwards, we will proceed with this new dataset.

In order to further improve our results, we have started to search by sector of activity. We separated our dataset into several datasets, the separation being done on the sector of activity of the companies. In doing so, we realized that if we applied different models for each business sector, we would get much better results. Depending on the operational needs, we conclude that this is an area to consider seriously.

Finally, we decided to look at the importance of features in our models. To do this, we looked at the importance of the variables in the classifiers, and we realized that only a fraction of the input variables were actually useful, and among those, our temporal features that we added. It would therefore be appropriate to reduce the number of input variables, and to go even further in temporalizing the model on the small remaining dataset.

Acknowledgements

For this internship, I had the opportunity to work for the CEDAR team of INRIA Saclay. I would like to particularly thank Mrs. Oana Balalau, my tutor in the company, who accompanied me throughout this internship. She brought me a great help, and was extremely pedagogical with me throughout this project. I would also like to thank the whole CEDAR team, and LabIA, in particular Mr. Robin Reynaud, without whom this internship could not have taken place. Indeed, my internship is the result of a partnership between the CEDAR team and the LabIA, which is a laboratory of Artificial Intelligence at the service of the government. Thanks to them, my internship could be carried out in the best conditions (considering the current health situation), with a follow-up and especially a regular contact (weekly with the CEDAR team, thanks to the Wednesday afternoons and the seminars).

In addition, I would also like to thank infinitely Mrs. Élodie Quézel and Mr. Simon Lebastard, who have welcomed me in their team at *SignauxFaibles*. I had the chance to work with them and join them in the office, and they were extremely understanding and educational with me. Having worked closely with Simon, I would especially like to thank him for his patience, and for taking the time to help me with the different problems we encountered, but also for helping me with the decision making, on which models to select, which steps to add, etc. Like Oana, I would like to thank her again for her support and help in these situations. Furthermore, I would like to thank Élodie, who welcomed me in the best possible way, who listened to me, and who was especially open to any proposal for pedagogy.

Finally, I would like to thank Mr. Vincent Viers and Mr. Chrisophe Ninucci from *SignauxFaibles*, who helped me a lot in my early days. They allowed me to get out of delicate situations very quickly, and they always knew how to take the time to help me in spite of their imperatives, and all this in a good mood.Working at their side was an enriching experience, although rare due to the sanitary measures.

Context and problem statement

During the five-year term of President François Hollande, one of the measures taken by the government was to help SMEs (Small and Medium-sized Enterprises), in order to prevent as much as possible the risks of failure. It is in this perspective that within the Ministry of Economy and Finance, teams have been dedicated to the identification of SMEs at risk of bankruptcy, and to the support of the latter. This is a relatively delicate mission, because in economics, it is quite important to let companies "die" so that new ones can appear. There is therefore a real balancing act, between saving the companies that have a high probability of prospering afterwards, and leaving those that could become so-called "zombie" companies, which only survive with the help of the government.

With this in mind, a team was created within the Direction Générale des Entreprises (DGE), and the state-owned start-up *SignauxFaibles* joined it. To be more precise, *SignauxFaibles* is a link in this chain. Its role is to develop predictive tools, which will enable the effective detection of companies at risk. Then, the various Commissioners for Restructuring and the Prevention of Business Difficulties (CRP) intervene within the targeted companies in order to provide them with the necessary support to avoid default.

Currently, *SignauxFaibles* offers a tool that ranks companies from most likely to fail to least likely. This forecast is done 18 months out, which means that if a company is listed as a bankruptcy risk, it will mean that in 18 months it has the potential to fail. To do this, they use a number of data that come from several different sources. These data are as follows :

— Debts on social contributions (monthly)
— Accounting and financial ratios (annual)
— Late payments to suppliers (monthly)
— Demand and consumption of partial activity (monthly)

This represents about forty entries (features). Once these data are collected, a pre-processing part is applied. This consists of a OneHoteEncoding of the textual variables (this is the vectorization of the textual data), followed by the addition of certain lag variables (for example, for a given column, we will add the values of this column over the last 6 months), and finally a normalization of the data. Following this step, a logistic regression is applied, which allows us to classify the companies in a binary way : survival of the company or death of the company. Then, a post-model processing step is performed by experts, in order to make the model a little more accurate than this simple binary classification. Following a decision tree, the experts will classify the companies in :

— Red for a high risk
— Orange for a moderate risk (e.g. it has been classified as a default risk by the model for several months, but its indicators are improving)
— Green to signify an absence of risk of bankruptcy

To evaluate this model, the teams decided to use several metrics : Balanced Accuracy, F_2-score, Precision, Recall and AUCPR. The Balanced Accuracy and the F_2-score will be our main metrics, and the other three metrics will serve as complementary indicators for our decision making in case models are equivalent on these two measures. The reason we select these two metrics primarily is that they contain within them the information from the other metrics. Balanced Accuracy is a metric that measures the performance of a binary classifier, in the case where the data is not balanced. This is our case, as we estimate that only between 5 and 10% of companies are at risk of default. The F_2-score is a harmonic mean of Precision and Recall, which places more importance on Recall than on Precision. Precision is a measure that calculates

the ratio of elements that are correctly classified as belonging to the failure class divided by the number of elements that are classified by the model as failure. The Recall is a measure of the number of elements that are assigned by the model to the bankruptcy class divided by those actually belonging to this class. In our case, a high Recall means that the model is effectively labeling failing firms, i.e., that few firms predicted as non-bankrupt are actually bankrupt.In the end, it will be up to the team to choose what is more important to them : a model with high Recall and potentially low Precision, which would mean that we would detect many firms that actually fail, but also many firms that are not failing but yet are predicted to fail ; or a model with high Precision and potentially low Recall, which would mean that we don't detect all failing firms, but we also don't get many nonfailing firms wrong. This decision is a function of several aspects, first and foremost the number of people available to handle failing firms. Finally, the AUCPR, which is the Area Under Curve, measures the benefits and costs of varying the Precision and Recall. Indeed, sometimes we will have a Precision that will decrease and a Recall that will increase, but it may be difficult to interpret this variation in terms of contribution, and this is where the AUCPR is useful, it quantifies it. AUCPR is nothing more than the area under the curve of the True Positive rate versus the False Positive rate. For clarity, here are the other formulas :

— Balanced Accuracy = $\frac{\frac{TP}{TP+FN}+\frac{TN}{TN+FP}}{2}$
— Precision = $\frac{TP}{TP+FP}$
— Recall = $\frac{TP}{TP+FN}$
— F_2-score = $(1+2^2)\frac{Precision*Recall}{2^2*Precision+Recall}$

where TP represents firms correctly classified as bankrupt, FP those wrongly classified as bankrupt, TN those correctly classified as not bankrupt and FN those wrongly classified as not bankrupt.

Currently, the model used by the team provides a Balanced Accuracy of 65.9%, an F_2-score of 0.17, a Precion around 75% and a Recall around 45% (these last two are averages over a number of trials). **The goal of my internship was to improve these results**. For that, several axes were evoked. **First of all, we made a benchmark of different models**, based on a specific bibliography. Indeed, the bibliography provided us with interesting results but not harmonized because the models were applied on different data sets. For this reason, we decided to apply all the models that we considered relevant to our dataset. **Next, one of the important axes to consider was the taking into account of temporality**. Indeed, the model was considered as too static, and it was thus envisaged to add this temporality, and to measure the contribution of this addition. **Then, in a more personal way, it was proposed to look sector by sector, which model would be the most relevant, in order to improve the results a little more**. **Finally, we will look at the importance of each of the features in the models where it is possible, in order to help the CRPs in their subsequent work**.

Improvement of the failure prediction model

1 Related Work

In order to start on a solid basis, we decided to start researching and reading several research articles, in order to be able to focus our project on a first path. We managed to find 4 articles that could be useful to us. How did we choose them ?

First of all, we had to take into account the constraints of the problem. Here, our goal is to provide a list of companies to people who do not necessarily have knowledge or even confidence in Artificial Intelligence and more specifically in Data Sciences. It was therefore important to create this trust. For this, the explainability of the model is a major criterion to take into account. So-called "black box" models were to be avoided as much as possible. Secondly, we wanted models that would improve our results, so models that performed well according to the different metrics used. And finally, the volume of data used and the temporal complexity. We want to have robust models, which train on data sets of sizes around one million entries, and whose complexity makes the training not too long.

Thus, we have chosen the following 4 articles :

— *Systematic review of Bankruptcy Prediction Models - Alaka et al, 2015* [1] : Comparison of several models, advantages and disadvantages of each
— *Bankruptcy prediction using imaged financial ratios and convolutional neural networks* [2] : Use of Deep Learning (CNN)
— *Anchors : High-Precision Model-Agnostic Explanations* [3] : Adding explainability to the model
— *White-box Induction From SVM Models : Explainable AI with Logic Programming* [4] : Explainability for SVM only

1.1 Systematic review of Bankruptcy Prediction Models - Alaka et al, 2015

In this article, several methods are tested, on different datasets, and these models are evaluated according to several criteria. Among these criteria, we find the three we need (explainability, performance and execution time). There are 5 others in addition, which are the following (ranked by importance) :

— **Multicollinearity/Correlation of variables** : measures the sensitivity of the model to the possibility of having features that are collinear
— **Assumptions needed for the model** : measures the initial constraints that must be met by, for example, the input data
— **Variable selection** : the selection of useful variables to optimize the model performance
— **Robustness to the non-homogeneity of the distribution** : this is useful here because the distribution of our variables is heterogeneous (remember that here there is a strong heterogeneity due to the fact that there is only a small part of the firms that go bankrupt)
— **Propensity to over-fitter** : here it will not be too much the case because our volume of data remains modest

The models we have chosen are : logistic regression (LR), support vector machine (SVM), decision tree (DT) and finally the multi layer perceptron (MLP). Let's detail what each of these models does.

Logistic regression is a conditional probability model, which uses the sigmoid function to calculate this probability. This model is a classical binary classifier which returns a probability between 0 and 1 which translates the probability that a company goes bankrupt. By default, if it is above 0.5 we will assign bankruptcy as an output, but we can modify this threshold.

The support vector machine uses a linear model to obtain an optimal separation of hyperplanes. It will construct the boundaries of these hyperplanes through binary classification. The variables closest to these hyperplanes will be called support vectors and will be used to determine the output value (outcome, in this case bankruptcy or not).

The decision tree is an iterative model that works by dichotomy. The decision tree will allow to provide decision rules afterwards. To know in which order to perform these rules, we will measure the importance. For example, if the turnover turns out to be more important in the classification than the number of employees, then the rule on turnover will be placed upstream.

The multi-layer perceptron is the basic model of neural networks. It has several layers that are fully connected to each other. This aims to reproduce the functioning of the human brain, hence the term neural networks.

Now we will list the results obtained in the table **1** :

Model	**Average score**	**Average FP %**	**Average FN %**	**Average F_2-score**	**Explainability**
LR	80.5%	18.87%	29.44%	78.77%	Trivial
SVM	83.0%	13.55%	26.22%	74.51%	Trivial if linear kernel
DT	80.5%	NA	NA	NA	Trivial
MLP	84.0%	14.47%	17.55%	84.89%	Black Box

Table 1 – Summary Table of Models Used

Nevertheless, this article has some limitations. First of all, we can wonder if, for each of the methods used, a cross-validation has been applied or not, and if for the neural models the temporal character is effectively taken into account. This would be of great help in our case. Moreover, for most of the studies included, we notice that they make the hypothesis of a balanced dataset (37 studies take proportions between 50% bankruptcy and as much non-bankruptcy and 30% bankruptcy and 70% non-bankruptcy, only 2 studies with proportions in 25-75% and still 2 around 15-85%, and only one with 0.3-99.7%). For studies with balanced datasets, we would have to compare the results of the review with our results after oversampling to find a balanced dataset. On the other hand, it is convenient for us that the unbalanced studies use our models.

1.2 Bankruptcy prediction using imaged financial ratios and convolutional neural networks

This article was of particular interest to us because it used sophisticated Deep Learning methods, and that was one of our goals. Let's detail what this article brings.

First, let's explain the reason for using Convolutional Neural Networks (CNN). CNNs have allowed us to obtain much better results (in terms of performance) than conventional methods. Moreover, CNNs are more adapted to image-related problems. Therefore, in order to use a CNN, it was necessary to transpose our numerical data into images.

Now let's talk about the database used. The study focuses on the companies of the **Tokyo Stock Exchange**. The data come from the **Nikkei NEEDS Financial QUEST database**, having for date of end June 2016, and taking the last 4 values for each company (the collection of the data is not regular, it is possible that for a company we have the data only quarterly, or that a month we do not obtain them, it is for that that we take the last 4 values obtained and not the values on the 4 preceding periods). In total, there are 175 input features from the balance sheets and 88 features from the P&L, which makes a total of 283 input features. A first filter is applied, in fact, we will only keep the features which will be effectively present in p% of the companies (in this case, p is 80 here). After that, the total number of features is 133.

Afterwards, we have 102 firms that failed and 2062 that did not, for a failure rate of 4.71%. Among the bankruptcies, we separate them into 5 sets of 20 firms, and we leave the last two aside. In the same way, we make 5 sets of 20 for the non-bankrupt firms and we leave the 1962 others aside. We transform these different sets into images. We create synthetic data using the weighted average method. We then draw 4 sets among the 5 for the two types of companies (there are thus 5 possible choices of training set/test set pairs). For the training set, we have 50% of bankruptcies and 50% of non-bankruptcies, bringing to 7520 x 2 images for the training set. For the test set, we have 88 images for the bankruptcies and 7928 for the non-bankruptcies. In other parts of the paper, they decide not to generate synthetic data for the non-bankrupt firms (they are already over-represented) or not to respect the 50-50 proportion in the training set. The results are different but not necessarily better.

For the dataset production we have explained how we distribute the data, and which data we keep. Let's detail the process by which we transform our digital data into images. We will apply two methods : a random method, which will randomly assign to the items a pixel on the image of size $N^{\frac{1}{2}}$ x $N^{\frac{1}{2}}$ where N is the number of items ; a correlated method, which will assign to the correlated ratios "close" pixels. Both methods are sensible, because for the random method, a CNN with enough layers will be able to determine patterns even between distant pixels, and for the correlated method, the pattern will be discernible with fewer layers. To begin, we randomly place the ratios on the image. Thanks to this, we will have generated the images of the Random method. We calculate the cost function (energy) associated with this image :

$E = \sum_{(i,j)} |c(R(i), R(j))| * d(i,j)$

$d(i,j) = [x(i) - x(j)]^2 + [y(i) - y(j)]^2$

where $R(i)$ is the financial ratio i, $c(R(i), R(j))$ the correlation coefficient between ratio i and ratio j and $(x(i), y(i))$ the coordinates of pixel i. If E can be reduced by exchanging two pixels (we swap them), then we make this change, otherwise, we do not change anything. We repeat this process, if there is no reduction of E after $3N$ steps, we stop. Once we have done this, we will generate synthetic data. To do this we will "average" the data from the existing data, and we will assume that by averaging such data, the synthetic firm will remain in the same class (bankruptcy or not) as the two firms used. The fact that we simulate synthetic data comes from the fact that we do not have enough values for the bankruptcy category, and for this reason we artificially create companies that will be.

The architecture used thereafter is the GoogLeNet, which is a CNN with 27 layers, and 7,000,000 input features, which can be seen in figure **1**.

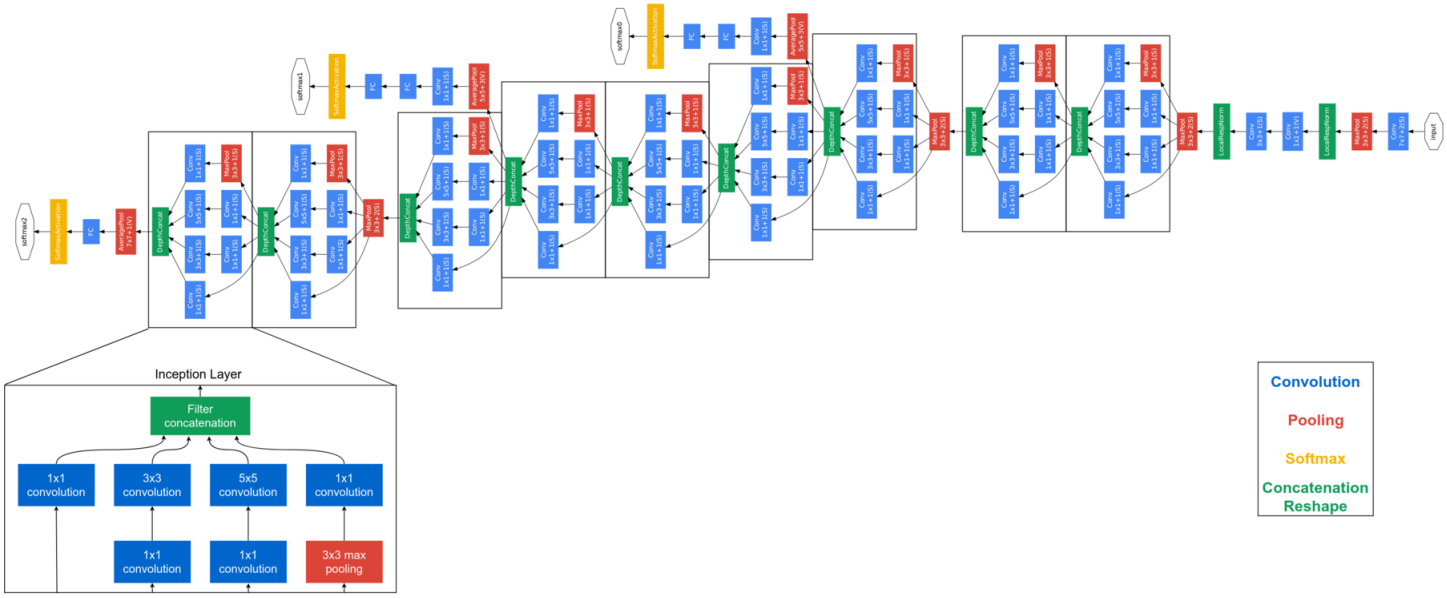


FIGURE 1 – Architecture of the GoogLeNet [6]

The limitations of this article are various. First, the complexity of the architecture used. Deep neural networks require more resources for training and prediction. Moreover, these models are not explicable. It would therefore be necessary to add an explanatory layer in the pipeline, making the model even more complex.

1.3 Anchors : High-Precision Model-Agnostic Explanations

In this article, the author will present an algorithm to explain "black box" models, such as the one described in the previous article [3].

The problem with explanatory models is that they are too local (e.g. in "it's not too bad" the "not" is positive, in "it's not too good" it is negative, and most explanatory algorithms fail to make this distinction). The goal is therefore to overcome this problem. The Anchors model relies on intuitive rules to explain the prediction of the model.

The goal of this algorithm is therefore to find the anchor (a kind of rule set) that will satisfy the accuracy constraint with a high probability. If several anchors are suitable, we will take the one that will cover the largest field (the set of rules that will apply to the largest number of inputs). This brings us back to an optimization problem.

The algorithm works like this : first, the set of rules A is empty (this rule applies to all inputs), then we add ONE rule, which gives us a set of rules at step 1. To find this rule, we generate one rule per feature, which gives us a set of new possible rules. Among all these rules, we take the one that satisfies the condition said in the previous paragraph. We repeat this process. This will give us the smallest anchor, but will not give us the "coverage" of the latter, i.e. we will not know its scope.

This greedy approach has some qualities but also some flaws. The first one is that it only allows an increment of one rule by one rule at each step, so any choice that would not be optimal would have heavy consequences on the continuation. Moreover, it does not seek to satisfy the "coverage" condition, but rather the notion of the smallest set of rules. Another approach is therefore necessary if we want to respect these two criteria. This approach is similar to greedy, except that instead of keeping THE best at each iteration, it keeps a predefined number B. Among these, it will choose the one with the highest coverage. We are therefore more likely to respect this second criterion.

In terms of results, we compare LIME to Anchor, and we look at the accuracy and coverage. LIME is an algorithm that disturbs the input values and looks at the consequences on the prediction, and thus draws conclusions on the functioning of the model. We compare 3 methods : a logistic regression, a neural network with 2 layers and 50 units each and 400 gradient boosted trees. We are going to apply them to classification at 3 levels : firstly, according to data on adults, predict their income (more or less than 50k/year), secondly, predict whether an ex-convict/person who has had dealings with the law will be sentenced again, and finally, the quality of a loan for a person according to financial data on them. For the three applications and the three models, the Anchor model gives us a much better accuracy than LIME (about 97% versus about 70%). Nevertheless, in terms of coverage, we realize that the LIME is more interesting for the first two applications, while the Anchor will be better for the third. Nevertheless, we have on average a coverage of 12% for Anchor against 15% for LIME, so it remains in the same order of magnitude.

The limits of this model are : when we are at the limits, we can have very specific decision rules and therefore not very flexible to generalization or we can have rules that will enter in opposition ; and finally the weak coverage (only 12% to 15% of the predictions are explained, that is to say that we remain with 85% of unexplained predictions).

1.4 White-box Induction From SVM Models : Explainable AI with Logic Programming

The goal is to bring explainability to the SVM, so that the resulting rules are understandable, accurate and faithful to the model. To do so, we will measure the Accuracy, the Recall and the F_1-score (to measure the fidelity).

First, we look at the SHAP model. To do this, we will measure the importance of each feature in the model. For any i such that x_i is a feature of the model, we will apply the model to any subset S of features not containing x_i, then apply the model to this same subset to which we will have added x_i. We subtract the results and see what happens. We do this for all the subsets not containing x_i then we average and we obtain what we call the Shapley value of i. We do this for all the data samples, and all the features, and it gives us a matrix.

This is the first approach, now let's talk about the Shap FOIL approach, which is a bit more sophisticated. The SHAP model allows to determine a set of features that will lead to a certain decision of the model. The Shap FOIL model allows the following : if a set of features explains what leads to a certain support vector, then it allows to give an explanation for all features "similar" to this vector.

We compare this model to an ALEPH (which is a state-of-the-art algorithm in this ILP domain), applying them to SVMs on a dataset coming from the UCI which collects data such as heart rate, blood pressure, etc. We separate the dataset into several categories (8 categories). We apply a classical SVM, and the two other models. In 7 cases out of 8, we realize that Shap FOIL outperforms ALEPH. The fidelity of Shap FOIL is shown by its F_1-score which is close to 0.9, against 0.8 for ALEPH. And above all, the F_1-score remains always higher than 0.8 where ALEPH's one reaches 0.55 in one case. In addition to that, the Shap FOIL model produces less rules than ALEPH, which is a big advantage in the explicability and its understanding for the user (we are around half less rules for Shap FOIL than for ALEPH).

2 Benchmark of the different models

As said before, we decided to restrict ourselves to a benchmark of some models. We kept four of them from [1], and we decided to add two others : the Random Forest, which will simulate N Decision Trees at the same time and will apply the principle of voting for the outcome, and the Gradient Boosting which is an ensemble architecture that consists in aggregating models (here they are Decision Trees) sequentially by weighting the training samples, and these same weights are modified during the training. Finally, we decided to look at the Voting Classifier of these six models. To explain simply, it will apply simultaneously the six models proposed above, and will apply a vote on the outcomes. Two choices are possible, either we make a weighted vote, and therefore we have to assign weights to each of the models, or we make a majority vote without any particular weighting (all models are equivalent).

In the whole suite, unless otherwise stated, we have a train set of 8.4 million entries, separated into 80% for training and 20% for validation. Similarly, we have a test set of 1.2 million inputs. The pipeline is as follows : first, we apply a OneHotEncoding on our textual features, then we normalize our data, and finally we apply the chosen model. Finally, we perform a Grid Search. The latter allows us to optimize the hyperparameters (for example for a Decision Tree, an interesting hyperparameter is the depth of the tree) of the model according to a chosen metric (here we have chosen the Balanced Accuracy). It is important to specify that this optimization is only partial. Indeed, the Grid Search will only perform its search on a (non-exhaustive) list of values that the user fills in himself.

2.1 Logistic Regression

We decided to use Logistic Regression for two reasons. First, in order to improve the existing model (which is itself a logistic regression). And second, in order to have a classical binary model that will serve as a control for the future.

For this model, we have applied a Grid Search on two hyperparameters : the penalty (l1, l2, elasticnet or none), and the coefficient C which will amplify or not the relugarization of the model.

Following this step, the best hyperparameters to optimize our balanced accuracy were : a penalty in l2 norm, and a coefficient C worth 1000 (as it is the inverse that counts, this large value reduces the importance of the regularization).

To identify the value of this model, we looked at the metrics we discussed earlier, but we were interested in another metric, which we will use in all our tests later. We decided to look at the Area Under Curve (AUCPR). This metric is interesting because it represents the area under the curve (the curve that plots the True Positive rate against the False Positive rate). This curve is a complementary value to Precision and Recall. It allows us to measure the compensation between the two, i.e. whether or not it is more interesting to decrease one by increasing the other or vice versa.

In the table **2** is a summary of all the metrics for this model. We notice that compared to the current model, we have an improvement in two of the main metrics. We gain 1.45% in balanced accuracy, and 0.225 in F_2-score. Nevertheless, this improvement remains relatively small for the first metric. The objective, if we follow Alaka et al. [1], would be to obtain results (at least) higher than 70%.

If we look at the other metrics, we have Precision and Recall that are in line with what the current model already offered. For the AUCPR on the other hand, we have decreased (by about 0.1). We thus realize that this model we propose is certainly an improvement for two of our reference metrics, but is not a unanimous candidate. It is therefore worthwhile to continue the research.

Model	AUCPR	F_2-score	Balanced Accuracy	TP	TN	FP	FN	Precision	Recall
Actual Model	0.6	0.17	65.9%						
LR	0.496	0.395	67.35%	1 492	89 330	481	2 742	75.62%	35.24%

TABLE 2 – Summary of Metrics for Logistic Regression

2.2 Decision Tree

One of the reasons for choosing the Decision Tree was its explicability, and the possibility of extracting a chain of rules intelligible to users.

For the hyperparameters, we are therefore interested in only the two most important ones : the depth of the tree (the deeper it is, the more rules there will be, and consequently the less interpretable it will be for the user because it will be too complex) and the criterion (this is the function that allows us to measure the contribution of the addition of a new rule).

Here, we have arrived at an interesting case. From a certain depth, we realized (see figure **2**) that the Test Accuracy (the one we are interested in) remained relatively constant (small increase) when the depth of the tree increased, before reaching an over-fitting depth of 12. To maintain a relatively good explainability, we therefore limited ourselves to a depth of 5.

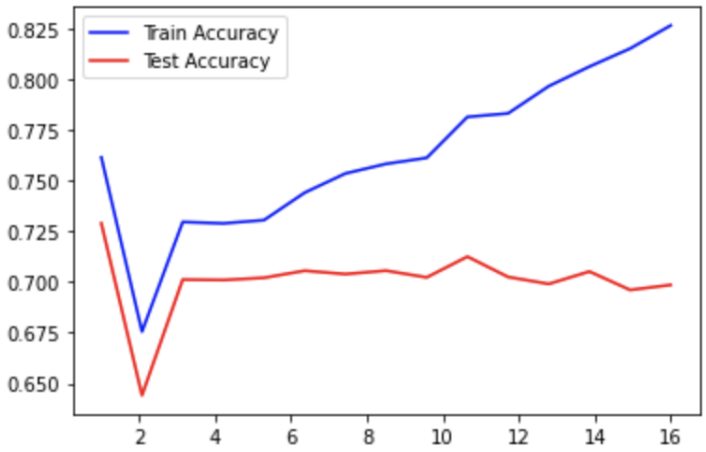


FIGURE 2 – Train Accuracy and Test Accuracy as a function of shaft depth

In the table **3**, we have listed the results. We can quickly see that the Decision Tree outperforms the Logistic Regression and the current model on the two main metrics. With a gain of nearly 3% for the Balanced Accuracy and a growth of nearly 400% for the F_2-score, it seems that this model is for the moment the best proposed. Moreover, its explainability makes it a strong candidate.

Nevertheless, we have some reservations, especially since the AUCPR has only slightly increased compared to the best results obtained with the current model. In addition, the Precision is much lower. That said, the Recall has increased, but does not exceed the best results obtained by the current model (which can reach up to 50%).

2.3 Support Vector Machine

The Support Vector Machine is a case apart. We ran into a major complication : its complexity. It was $O(p * n^2 + n^3)$, where p is the number of features (here about 50) and n the size of the dataset. When

Model	AUCPR	F_2-score	Balanced Accuracy	TP	TN	FP	FN	Precision	Recall
Actual Model	0.6	0.17	65.9%						
DT	0.529	0.467	71.06%	1 818	89 080	731	2 416	71.32%	42.94%

Table 3 – Summary of Metrics for the Decision Tree

we ran it on the whole dataset, it took several days to finish. So we decided to limit ourselves to 100 000 rows for the training set, 20 000 rows for the validation set and 50 000 rows for the test set. Moreover, we limited ourselves to a Linear kernel to save execution time.

For the hyperparameters, we limited ourselves to the regularization coefficient and the γ coefficient. The latter configures the sensitivity to the differences between the support vectors. A too large γ can tend to over-fitting.

Finally, the optimal coefficients were relatively low, reflecting a strong desire to regularize and not fall into over-fitting.

In the table **4**, we can easily see that this model is not very useful for our application. Its reference metrics are not very encouraging. With a Balanced Accuracy roughly equal to that of the current model and a Recall that is significantly lower, it would seem that this model is not worth keeping. Moreover, its other metrics are not very interesting either. None of the metrics seem to be the best. Finally, if we go back to the complexity of the model, we can conclude that this model should be avoided for our field of application.

Model	AUCPR	F_2-score	Balanced Accuracy	TP	TN	FP	FN	Precision	Recall
Actual Model	0.6	0.17	65.9%						
SVM	0.503	0.367	66.05%	560	48 105	169	1 166	76.82%	32.44%

Table 4 – Summary of Metrics for Support Vector Machine

2.4 Random Forest

The Random Forest joins the other models that use all our dataset data. It is nevertheless one of the models whose complexity and computation time are the longest. Indeed, its complexity is $O(p * n_{trees} * n^2)$ where p and n are the same as for the SVM, and n_{trees} is the number of trees used.

For this model, we are interested in three hyperparameters. First, the depth of each tree. In the same way as for the Decision Tree, the depth of each tree will have an impact on the notion of over-fitting. Moreover, we have also kept the criterion, which still allows us to measure the contribution of each new rule at each step. And finally, the max features. The latter is used to select the number of features to consider for each new split. For the sake of complexity, we have kept the number of trees at 100, otherwise we would fall into a situation similar to that of the SVM.

Given the metrics in the table **5**, it is clear that this model outperforms the current model. With a gain of 3.3% for the Balanced Accuracy and a growth of 250% for the F_2-score, the Random Forest is to be considered. However, with respect to these two metrics, which are the most important for us, the Decision Tree remains the most interesting. Moreover, its Recall is too low compared to the expected requirements.

Nevertheless, for all the additional metrics, we realize that the Random Forest is the best candidate. It is therefore necessary to take it into account and seriously. It is surely possible to improve its Balanced Accuracy and its F_2-score.

To summarize, we are dealing with a model that is very interesting, but more time consuming than the other models we have considered so far.

Model	AUCPR	F_2-score	Balanced Accuracy	TP	TN	FP	FN	Precision	Recall
Actual Model	0.6	0.17	65.9%						
RF	0.599	0.432	69.20%	1 649	89 317	494	2 585	76.95%	38.95%

Table 5 – Summary of Metrics for the Random Forest

2.5 Gradient Boosting

For Gradient Boosting, other hyperparameters must be taken into account. Indeed, the notion of learning rate appears for the first time. The latter is used when we update the weights associated with each of the sub-models used.

Other hyperparameters we are looking to find to optimize our Balanced Accuracy include the depth of our trees, the number of trees we simulate, and the criterion. These are parameters that we found in the other two decision tree models, with the number of trees added.

We realized here that only 10 trees were needed to obtain an optimal score. Moreover, the depth of the trees used was very low, a depth of only two was enough. This is quite surprising, but considering the results obtained, it seems more than satisfactory.

Here, we have the best model regarding the Balanced Accuracy and the F_2-score. Its execution time is however more important than the Random Forest. To give an order of magnitude, we had to run it over several days to get interesting results on the whole dataset.

As for the other metrics, they are much less interesting. For the most part, we have the worst results so far, except for Recall. But Recall is quite an important metric for this application. Indeed, it measures the ability of a model to detect True Positives (here TP). This is what is important for us here. The fact that it labels a little more False Positives is not too important (as long as this "a little" remains low), because the main goal here is to provide **all** (or almost all) the companies likely to fail.

Model	AUCPR	F_2-score	Balanced Accuracy	TP	TN	FP	FN	Precision	Recall
Actual Model	0.6	0.17	65.9%						
GB	0.310	0.477	71.84%	1 900	88 731	1 080	2 334	63.76%	44.87%

Table 6 – Summary of Metrics for Gradient Boosting

2.6 Multi Layer Perceptron

The Multi Layer Perceptron is the only Deep Learning algorithm we will use in this study. Indeed, we wanted to avoid as much as possible the "black box" models, because they are not simply explainable.

The aim of this test was to measure the contribution of Deep Learning on our results, and thus to determine if it was interesting to use it, even if it means adding an explanatory part to the pipeline.

In this classical Deep Learning model, we were interested in several hyperparameters. First of all, the activation function. It is this function that will allow us to move towards non-linear models. Then, the number of layers in our neural network. In the same way as for decision tree models, we must not put

too many layers to avoid overfitting, or even simply too much complexity in execution. Then, the learning rate. We wanted to determine whether we should have a constant rate or not, and what the initial value of this rate should be to maximize our Balanced Accuracy. And finally, we also wanted to look at the tolerance. Tolerance is used to stop the optimization as soon as the metric under consideration does not increase sufficiently.

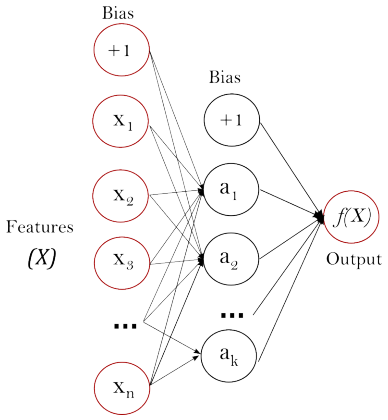


FIGURE 3 – Explanatory diagram of a Multi Layer Perceptron [5]

In our case, the optimal number of layers was 18. This seems quite consistent, since we had about 40-50 features, so we did not expect to find a number higher than that. The selected learning rate should be constant and equal to 0.005, and the tolerance to 0.001.

In the table **7**, it is clear that the Multi Layer Perceptron is the model that offers us the highest Balanced Accuracy. Its F_2-score is not particularly spectacular, although it is clearly better than the current model. Moreover, its execution time is shorter than the three previous models. Despite its non-interpretable side, it remains a serious candidate in terms of performance.

His other results are not necessarily exceptional compared to the others, they are in the general average. However, its Recall is the highest obtained so far. His ability to detect all True Positives is therefore the best of the new models.

Model	**AUCPR**	**F_2-score**	**Balanced Accuracy**	**TP**	**TN**	**FP**	**FN**	**Precision**	**Recall**
Actual Model	0.6	0.17	65.9%						
MLP	0.484	0.457	72.11%	1 918	88 833	978	2 316	66.23%	45.30%

TABLE 7 – Summary of Metrics for the Multi Layer Perceptron

2.7 Voting Classifier

We decided to use a Voting Classifier to see if this simple principle could improve our metrics.

To do this, we used all the previous models with the hyperparameters selected to optimize their Balanced Accuracy. We then had to find the way to carry out this vote. The hyperparameters to find were the type of vote (simple majority or weighted votes), and if the vote is weighted, then the weights associated with each of the models.

The best results were obtained for a weighted vote, giving the same weight to all the models except for the Random Forest, whose vote will have three times the importance. This seems quite logical, considering that it is one of the most interesting models.

In the table **8**, we quickly realize that this test is not very conclusive. The execution time is longer, and especially the results obtained are never indisputable. The main scores are not optimal compared to the others. This is probably due to the fact that we did not necessarily find the best weighting. Indeed, the Grid Search looks for the best hyperparameters among those provided, which is not an exhaustive list of possibilities. We therefore decided not to continue with this model and to focus on the few most relevant models.

Model	**AUCPR**	F_2**-score**	**Balanced Accuracy**	**TP**	**TN**	**FP**	**FN**	**Precision**	**Recall**
Actual Model	0.6	0.17	65.9%						
VC	0.580	0.447	70.01%	1 723	89 204	607	2 511	73.95%	40.69%

Table 8 – Summary of the Metrics for the Voting Classifier

2.8 Conclusion of this benchmark

In the table **9**, we have summarized all the values of the metrics for the different models, and we have highlighted in green the best values for each metric.

In view of what we have said above and what we see in this table, we realize that, apart from the SVM, the most complex models in terms of time are the most efficient. The Random Forest and the Multi Layer Perceptron are each the best models for two metrics. Moreover, if we take a more general interest, we realize that the Decision Tree is also a very good model, as it is consistently among the three best models.

As we have said in this section, some models are excessively expensive in terms of execution time and complexity. This is why we decided to penalize them a little more than the others. This is the case for the Support Vector Machine and the Gradient Boosting. For the rest of our study, we decided to use them to a lesser extent, in order to gain in computation time, but also for operational gain. The team of *SignalsFaibles* agreed on the fact that the gain in performance was not sufficient to make up for this time cost.

Model	**AUCPR**	F_2**-score**	**Balanced Accuracy**	**TP**	**TN**	**FP**	**FN**	**Precision**	**Recall**
Actual Model	0.6	0.17	65.9%						
LR	0.496	0.395	67.35%	1 492	89 330	481	2 742	75.62%	35.24%
DT	0.529	0.467	71.06%	1 818	89 080	731	2 416	71.32%	42.94%
SVM	0.503	0.367	66.05%	560	48 105	169	1 166	76.82%	32.44%
RF	0.599	0.432	69.20%	1 649	89 317	494	2 585	76.95%	38.95%
GB	0.310	0.477	71.84%	1 900	88 731	1 080	2 334	63.76%	44.87%
MLP	0.484	0.457	72.11%	1 918	88 833	978	2 316	66.23%	45.30%
VC	0.580	0.447	70.01%	1 723	89 204	607	2 511	73.95%	40.69%

Table 9 – Summary of Benchmark Metrics

For the rest of our study, which will focus on the consideration of temporality, we have decided to take only four models, the four most relevant and effective in our opinion :

- the Logistic Regression : which is our guiding thread and our buffer model
- the Decision Tree : which is a simple, fast, very interpretable and efficient model
- the Random Forest : which offers us very good results
- the Multi Layer Perceptron : which is very efficient and fast in execution, although not interpretable (but we decide to keep this neural model to measure the contribution of Deep Learning in our application)

3 Taking into account the temporality

One of the objectives we had set ourselves was to add a temporal character to our model. Indeed, for the moment we were looking at the values in a static way (except for a few features for which we had added the last six values).

We therefore decided to make a first temporal approach in order to measure the interest of this addition. The idea is the following. We took the whole dataset, and we removed all the variables called "lag" (these are the last six values). Then, we decided to add for all numerical fields the average values, and the relative variations with respect to this average value, the whole on a given period (we first tested on a 6 months period, but finally on a 12 months period the results were more interesting and relevant because some data are collected only in a yearly way).

Once this was done, we ran our previously selected models (Logistic Regression, Decision Tree, Random Forest and Multi Layer Perceptron) again, and compared them to the results obtained without this addition. Next, we separated our numerical features into several categories : **Financial Health**, **Partial Activity**, **Delayed Payment**, **URSSAF Debts** and then **Paydex** (the **Paydex** group contains a financial indicator which is a score between 1 and 100, which allows to reflect the ability of a company to repay its debts ; the higher the score, the better the probability that the company will repay). We made this separation in order to add only one category, and this for all categories, to measure the performance and impact of each category on the classification.

3.1 Comparison between the initial dataset and the fully modified dataset

We ran our four models again on these new datasets. First when the selected period was 6 months, then 12 months. We obtained new hyperparameters to optimize our Balanced Accuracy.

In the table **10**, we have listed the set of metric values for each model and dataset pair. In green, we have highlighted the pairs that overperform the original model and dataset.

First, we realize that for all the models, the dataset using a period of 6 months decreases the performance of the models. Indeed, except for the Precision which seems to increase, all the other metrics (on average) seem to decrease. We can therefore conclude that the choice of the 6-month period was not judicious. As we said, since some data are only collected every year, it was predictable that a 6-month period would not be optimal, because the model would learn too much about some features and not enough about others.

On the other hand, if we look at the models applied to the dataset over a 12-month period, we notice a clear improvement in results. For the Logistic Regression, we obtain the greatest increase in the Balanced Accuracy (+3.55%), the F_2-score (+0.076) and the Recall (+6.88%). For Precision and AUCPR, the Multi Layer Perceptron provides us with the biggest increases (+17.44% and +0.133 respectively). Finally, the model providing us with the best results for the main metrics we have selected (except Precision) is the Multi Layer Perceptron.

In general, we realize that this dataset over a period of 12 months is more than convincing, as it provides us with significantly better results on all the metrics used (except for the Decision Tree Recall).

Model and Dataset	Balanced Accuracy	F_2-score	Precision	Recall	AUCPR
Actual Model + original dataset	65.9%	0.17			0.6
LR + original dataset	67.35%	0.395	75.62%	35.24%	0.496
LR + dataset over a period of 6 months	67.23%	0.392	76.06%	34.98%	0.478
LR + dataset over a period of 12 months	70.90%	0.471	88.72%	42.12%	0.629
DT + original dataset	71.06%	0.467	71.32%	42.94%	0.529
DT + dataset over a period of 6 months	67.13%	0.390	74.06%	34.84%	0.479
DT + dataset over a period of 12 months	71.06%	0.473	87.27%	42.49%	0.561
RF + original dataset	69.20%	0.432	76.95%	38.95%	0.599
RF + dataset over a period of 6 months	67.89%	0.408	87.52%	36.03%	0.566
RF + dataset over a period of 12 months	71.75%	0.488	88.56%	43.85%	0.660
MLP + original dataset	72.11%	0.457	66.23%	45.30%	0.484
MLP + dataset over a period of 6 months	69.95%	0.444	69.80%	40.75%	0.503
MLP + dataset over a period of 12 months	72.54%	0.502	83.67%	45.61%	0.617

TABLE 10 – Summary of metrics for each model and dataset

3.2 Comparison between the initial dataset and the datasets where only the lag variables of a feature category are added

You will find the comparison tables in the Annexes. A first observation is the following : the results over a 12-month period are clearly better than those over 6 months, and than those compared to the original dataset. We obtain values for the Balanced Accuracy and the F_2-score that are nearly (respectively) 10% and 0.4 larger than for the model currently used. Precision will gain up to 15% and Recall will gain 5%. Therefore, adding the categories one by one is a real advance for our application. Now let's go into more detail.

3.2.1 Logistic Regression

For this model, we quickly realize (table **11**) that adding the categories one by one over a period of 6 months does not bring anything on some of the metrics that interest us. On the other hand, on Precision, we have a slight gain, but nothing exceptional. We can therefore conclude that this addition will not be useful to us and therefore we can leave it aside.

However, when we move to a 12-month period (table **12**), the results are better for each category. Moreover, we see that when we add only the category **Paydex**, we obtain better results for our two basic metrics, than when we add all the categories. For this model, it would therefore be interesting to continue the research, keeping the same idea that the minimum period necessary is around 12 months.

One last remark, which applies to the two other models that follow. The categories **Financial Health**, **Partial Activity**, **Late Payment** and **URSSAF Debts** give us exactly the same values for the metrics. So we decided to look in more detail to understand why. The reason is the strong correlation between these categories. In the category **Financial Health**, we have a very large number of features (about ten), and for the three other categories, we only have between one and three features. However, these three features are values closely related to the features of the category **Financial Health**. For example, for the URSSAF debt, we have a debt ratio column. This value is obtained directly from the turnover and the debt, which are found in the category **Financial Health**.

3.2.2 Decision Tree

As for the Logistic Regression, by the correlation between the first four categories, we have the same consequences for the results on these categories.

Here, we see in the table **13** that the results we get for adding the categories one by one over a period of 6 months are more interesting and better than if we add all the categories at once, for all the metrics. In addition, and most importantly, for the category **Paydex**, we have an outstanding gain in Balanced Accuracy. We go from 71.06% for the original dataset to 76.03%. This is our best result so far. On the other hand, the F_2-score remains average, which reflects a poor ability to detect the majority of companies that are actually failing. This is a point to improve if we want to keep this model and this dataset.

In the table **14**, the Balanced Accuracy and the F_2-score are not very interesting, because we get better results over the same 12-month period if we add all the categories at once. Nevertheless, we get an Accuracy that is close to 90%, which is excellent and means that when a company is labeled as failing, then we can be almost certain that it will, where with the original dataset we could not be.

3.2.3 Random Forest

In the table **15**, we realize that adding the categories over a period of 6 months one by one is advantageous in terms of F_2-score and Precision. For the rest, like Balanced Accuracy, we cannot draw an unambiguous conclusion, although for the category **Paydex** we do indeed obtain the best results.

Let's move on to the results in the table **16**, which become extremely different and indistinguishably better. For all categories, we have better results, and this compared to all our previous tests. With a Balanced Accuracy around 75.30% for all categories, a Precision of 91.05%, a Recall around 51% and the best F_2-score obtained so far (around 0.558), we can only be satisfied with this model and these datasets.

A first conclusion is that taking into account the temporality, in this way, for this Random Forest model is a real progress. Its only drawback would be its execution time, which takes a whole day, but the gain in performance is for once important enough to afford this time cost.

3.2.4 Multi Layer Perceptron

Here, compared to the others, because of the neural character, we will not obtain the same results for the first four categories. Indeed, the previous models are statistical models, as it is said in [1]. The neural models are more complex, and go beyond statistics, and that is why they are difficult to interpret. Moreover, the initial weights are initialized to a certain random value, potentially different for each feature, and therefore it is not necessary that the convergence is done towards a single value for all.

In the table **17**, we have results that are significantly better for the **Santé Financière** category over a 6 month period. Nevertheless, we have a F_2-score that remains relatively low compared to other models, and a low Precision. Given the other models, we cannot be satisfied with this.

If we look at the table **18** which is on a 12 months period, as for the Random Forest, the results are clearly better and this for all the metrics. The scores obtained are close to 75% for the Balanced Accuracy, 0.560 for the F_2-score, 85% for the Precision and 50% for the Recall. Once again, this line of research is very encouraging for the future. The only drawback of this model, and this since the beginning : its explainability.

3.3 Conclusion

In this part, we realized at the beginning that being interested in a dataset (with all categories or just one at a time) over a period of 6 months was uninteresting or even disadvantageous. On the other hand,

we realized that if we took a dataset over a period of 12 months, the results were very encouraging, and therefore this type of method was to be considered.

If we go into more detail, we were able to identify two models in particular. First, for trivial explainability, the Decision Tree applied to a dataset over a period of 12 to which we added that the category **Paydex**. It is for this model and this dataset that we have obtained the optimal Balanced Accuracy so far, and an excellent Precision and a correct Recall, which translates into a good F_2-score.

Then, we will favor the Random Forest applied on a dataset over a period of 12 months to which we have added the category **Paydex**. It is for this model and this dataset that we obtain the best F_2-score, and an excellent Balanced Accuracy.

4 A model by sector ?

In this part, we decided to see if we could increase our performances by proposing models by sector of activity. It is good to specify that we used as dataset the previously computed dataset, i.e. we use the **original dataset to which we added the temporal features over a period of 12 months**.

To do this, we applied the four models that we selected at the end of the benchmark to each of the sectors of activity. The results obtained can be found in figure **7**, and the correspondence between the codes of sector and the name of the sector is to be found in Annexes in figure **8**.

For some sectors, notably the sector "Agriculture, forestry and fishing" of code "A", we see that all the statistical models provide us with a Balanced Accuracy of 100% and the neural model a Balanced Accuracy of 50%. It is worth noting that for these sectors, we have very little data. Indeed, we are mainly interested in industrial sectors, as our data sources come from these areas. Therefore, for the category mentioned here, we had only 500 values in the training dataset for 38 values in the bankruptcy category, and 79 values in the test dataset of which only one company was in bankruptcy.

For the sectors that interest us the most, we notice that for 11 of them, the most interesting model is the Multi Layer Perceptron and in second place the Random Forest, which surprises us very little considering that these two models were the ones we had selected in terms of performance. On the other sectors, it is the Random Forest that wins. We therefore remain in this logic, according to which our two models, which are very efficient in the global sense, remain so when we go into the granularity of the sectors of activity.

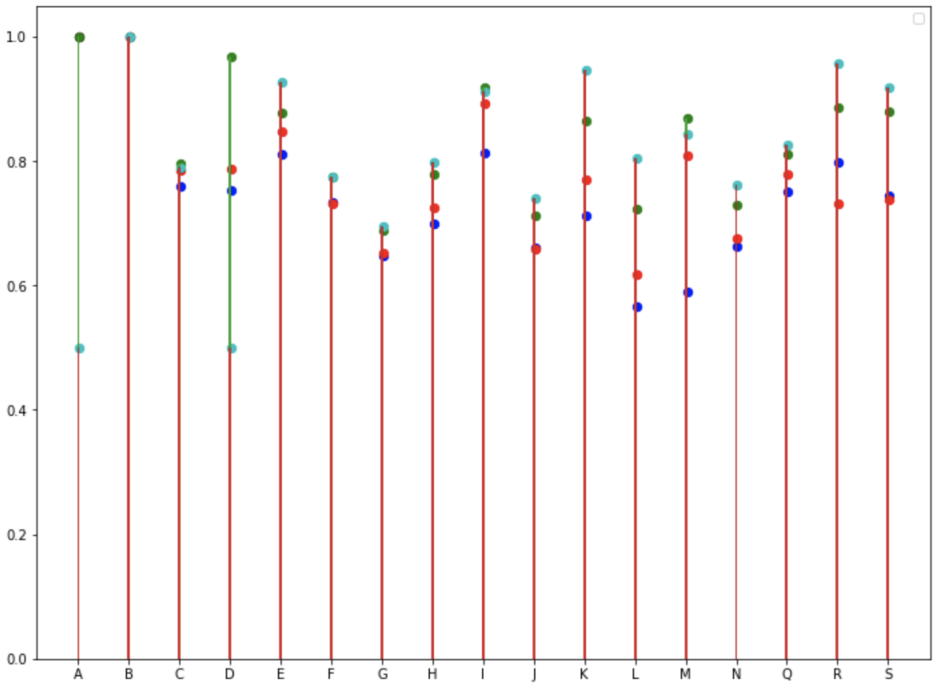


FIGURE 4 – Balanced Accuracy by model and sector code
blue : Logistic Regression : red : Decision Tree
green : Random Forest : cyan : Multi Layer Perceptron

Nevertheless, we realize that in certain sectors, such as the **Arts, Entertainment and Recreation** and **Real Estate** sectors (with respective codes **R** and **L**), the fact of selecting the Multi Layer Perceptron rather than the Random Forest allows us to gain between 5% and 10% in Balanced Accuracy, which is not negligible

5 The importance of features

In this part, we decided to look at which features, i.e. which data had the most importance in the prediction of each of the statistical models (we cannot access this information for the Multi Layer Perceptron because of its neural character, as the weights we would obtain are those obtained for a given layer, which does not give us precise information since between each layer there is a feature transformation).

5.1 Logistic Regression

For the Logistic Regression, we realize thanks to the figure **5** that it is the features with a rank between 60 and 100 that matter the most, the others being negligible. If we look in detail, it is the set of numerical data (thus the categories that we used in the Temporality part : **Financial Health**, **URSSAF debts**, **Delayed Payments**, **Partial Activity** and **Paydex**), and on top of that, the temporal features for the category **Financial Health**. The two most important features in the prediction are : the **employer's share amount** and the **average value of the worker's share amount over 12 months**. It is interesting to realize that finally our consideration of temporality has had such an impact on the model, and this confirms the interest of continuing on this path.

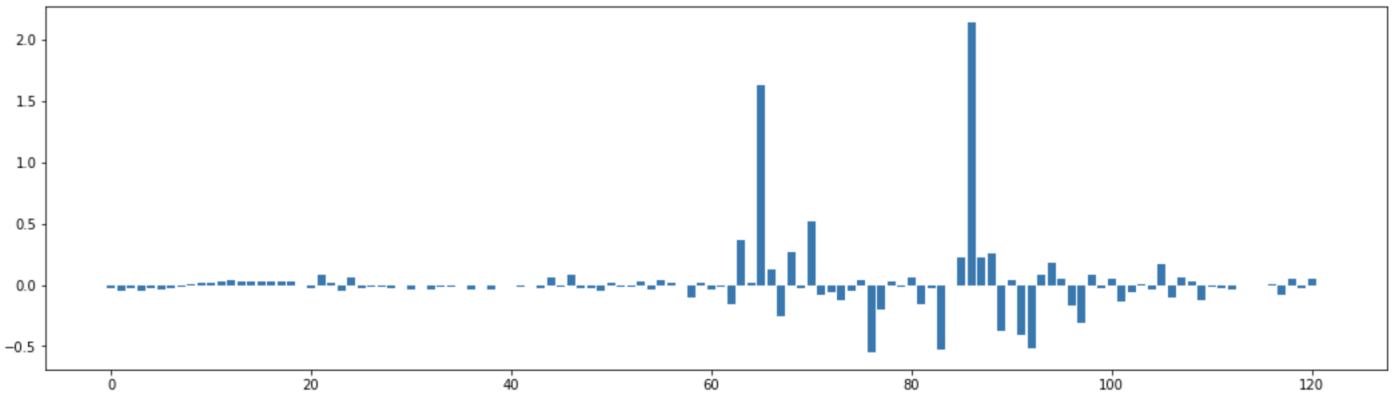


FIGURE 5 – Features Importance for Logistic Regression

5.2 Decision Tree

For the Decision Tree, we have a very interesting result in figure **6**. We have a column that largely predominates and very clearly the others, and a second one that matters but almost eight times less than the first one, and the others contribute almost nothing.

When we look at what these two fields correspond to, we realize that they are the same as for Logistic Regression. This confirms our interest in taking temporality into account, and we realize that we could even limit ourselves to fewer fields and perhaps deepen the temporal aspect on the few fields we have left.

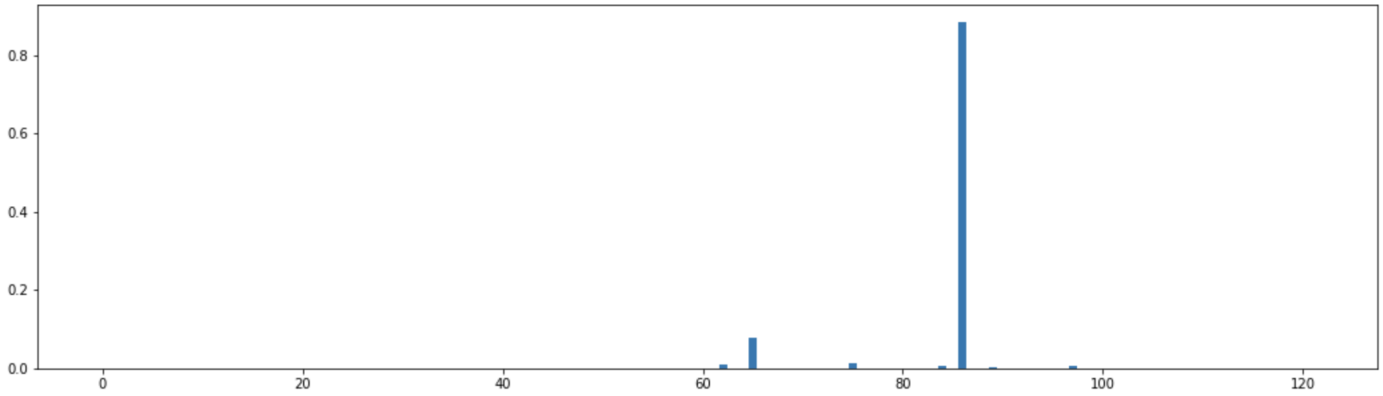


FIGURE 6 – Features Importance for the Decision Tree

5.3 Random Forest

For this model, in figure **7** we see that we have the same conclusions as before. The important features are the numerical ones, with a strong predominance for four of them : the same as the previous ones, the amount of the worker's share and finally the average over 12 months of the contributions. Once again, we realize the value of having added a temporal aspect to our models.

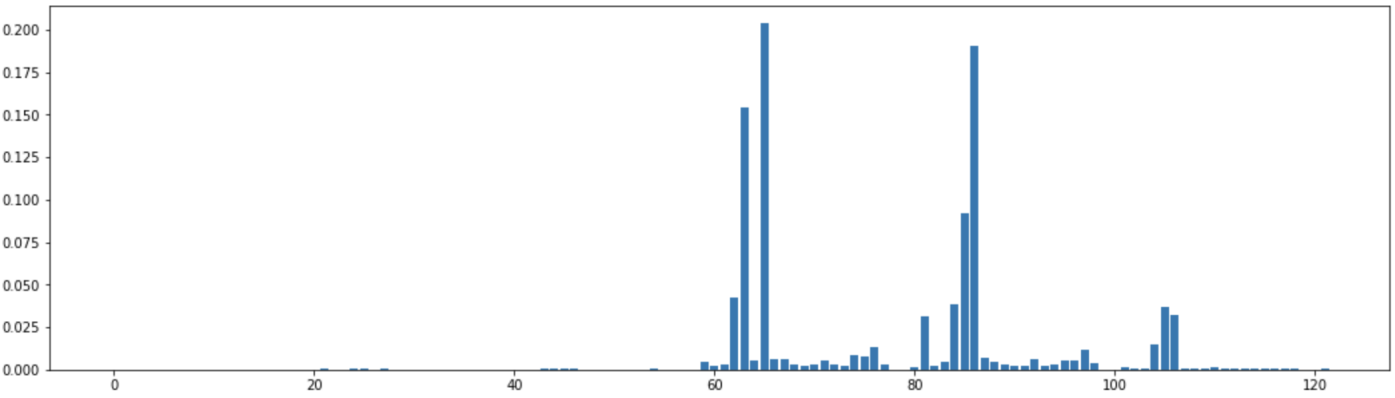


FIGURE 7 – Features Importance for Random Forest

5.4 Conclusion

Thanks to this small study of the importance of features, we have realized that the temporal aspect is to be taken into account very seriously, and it would be good to deepen it. Another interesting aspect is that it is not necessary to keep all our fields. This would be very useful to be able to go deeper into the temporality of the model, by adding more temporal variables or temporal ratios for the few fields that are finally necessary and sufficient.

General conclusion

In this study, we were able to realize that we could easily improve the performance of the current model, while improving the explainability of the model.

Thanks to our benchmark of several models, we can conclude that in order to obtain better and more explainable results, it would be sufficient to use a Decision Tree type model. We would obtain better metrics for the same data as currently, and above all, we could easily obtain the chain of rules that allowed us to conclude whether or not the companies were bankrupt. It is also in this part that we could realize that other models, certainly not explainable, offered us much better performances, in a relatively correct computing time.

Nevertheless, the results of our benchmark were not sufficient, and to gain the trust of users, we needed to obtain more conclusive results. This is where our study on temporality was more than useful. It provided us with more than correct results. We could see that it was easily possible to obtain excellent performances by adding lag variables for some of our fields. This allowed us to obtain Balanced Accuracy close to 75%, an F_2-score of 0.550, a Precision of 90% and a Recall of 50%. It is therefore natural to conclude positively on this study, and it would be recommended to continue the efforts on this way.

Furthermore, once we had identified four interesting models in the first part and measured the importance of temporality in the second part, we applied these models to this new dataset, separating them by sector of activity. This allowed us to conclude that it would be even more relevant to separate the models by sector, as this could save us up to 10% in Balanced Accuracy.

Finally, our study on the importance of features allowed us to realize that only a handful of fields were actually useful in our models, and among these fields, the fields we had added in the temporal part. This confirmed our interest in our temporality contribution, and allowed us to conclude that we could limit ourselves to these "useful" fields, and to continue the temporal analysis on these fields.

One part of the study that we were not able to address due to lack of time is the explanation of certain models. For example, in view of the results for the Random Forest and the Multi Layer Perceptron, it would be judicious to measure the impact on the performance of adding an explanatory branch of the model to the pipeline.

Bibliography

References

[1] Alaka et al. *Systematic review of Bankruptcy Prediction Models*. 1993.

[2] Tadaaki Hosaka. *Bankruptcy prediction using imaged financial ratios and convolutional neural networks*. Tokyo University of Science, 1-11-2 Fujimi, Chiyoda City, Tokyo, Japan, 2018.

[3] Marco Tulio Ribeiro, Sameer Singh, Carlos Guestrin. *Anchors : High-Precision Model-Agnostic Explanations* 2018.

[4] Shakerin F, Gupta G. *White-box Induction From SVM Models : Explainable AI with Logic Programming* 2020.

[5] Scikit-learn.
`https://scikit-learn.org/stable/_images/multilayerperceptron_network.png`

[6] Nur Ateqah Binti Mat Kasim , Nur Hidayah Binti Abd Rahman , Zaidah Ibrahim , Nur Nabilah Abu Mangshor. *Celebrity Face Recognition using Deep Learning*
`file:///Users/tom/Downloads/Celebrity_Face_Recognition_using_Deep_Learning.pdf`,
Novembre 2018

Table of Figures and Tables

List of Figures

List of Tables

Appendix

LR + dataset over a period of 6 months for a category	Balanced Accuracy	F_2-score	Precision	Recall	AUCPR
Financial Health	66.91%	0.386	77.78%	34.29%	0.485
Partial Activity	66.91%	0.386	77.78%	34.29%	0.485
Delayed Payment	66.91%	0.386	77.78%	34.29%	0.485
URSSAF debts	66.91%	0.386	77.78%	34.29%	0.485
Paydex	66.85%	0.385	77.87%	34.15%	0.484

TABLE 11 – Summary of Logistic Regression Metrics for each category over a 6 month period

LR + dataset over a period of 12 months for a category	Balanced Accuracy	F_2-score	Precision	Recall	AUCPR
Financial Health	70.89%	0.470	88.80%	42.09%	0.630
Partial Activity	70.89%	0.470	88.80%	42.09%	0.630
Delayed Payment	70.89%	0.470	88.80%	42.09%	0.630
URSSAF debts	70.89%	0.470	88.80%	42.09%	0.630
Paydex	70.94%	0.471	88.79%	42.19%	0.628

TABLE 12 – Summary of Logistic Regression Metrics for each category over a 12 month period

DT + dataset over a period of 6 months for a category	Balanced Accuracy	F_2-score	Precision	Recall	AUCPR
Financial Health	69.20%	0.433	78.43%	38.91%	0.587
Partial Activity	69.20%	0.433	78.43%	38.91%	0.587
Delayed Payment	69.20%	0.433	78.43%	38.91%	0.587
URSSAF debts	69.20%	0.433	78.43%	38.91%	0.587
Paydex	69.36%	0.436	80.34%	39.17%	0.581

TABLE 13 – Summary of Decision Tree Metrics for each category over a 6 month period

DT + dataset over a period of 12 months for a category	Balanced Accuracy	F_2-score	Precision	Recall	AUCPR
Financial Health	75.27%	0.558	91.06%	50.84%	0.716
Partial Activity	75.27%	0.558	91.06%	50.84%	0.716
Delayed Payment	75.27%	0.558	91.06%	50.84%	0.716
URSSAF debts	75.27%	0.558	91.06%	50.84%	0.716
Paydex	75.37%	0.560	91.05%	51.05%	0.714

TABLE 14 – Summary of Decision Tree Metrics for each category over a 12 month period

RF + dataset over a period of 6 months for a category	Balanced Accuracy	F_2-score	Precision	Recall	AUCPR
Financial Health	69.20%	0.433	78.43%	38.91%	0.587
Partial Activity	69.20%	0.433	78.43%	38.91%	0.587
Delayed Payment	69.20%	0.433	78.43%	38.91%	0.587
URSSAF debts	69.20%	0.433	78.43%	38.91%	0.587
Paydex	69.36%	0.436	80.34%	39.17%	0.581

TABLE 15 – Summary of Random Forest Metrics for each category over a 6 month period

RF + dataset over a period of 12 months for a category	Balanced Accuracy	F_2-score	Precision	Recall	AUCPR
Financial Health	75.27%	0.558	91.06%	50.84%	0.716
Partial Activity	75.27%	0.558	91.06%	50.84%	0.716
Delayed Payment	75.27%	0.558	91.06%	50.84%	0.716
URSSAF debts	75.27%	0.558	91.06%	50.84%	0.716
Paydex	75.37%	0.560	91.05%	51.05%	0.714

TABLE 16 – Summary of Random Forest Metrics for each category over a 12 month period

MLP + dataset over a period of 6 months for a category	Balanced Accuracy	F_2-score	Precision	Recall	AUCPR
Financial Health	72.12%	0.484	66.94%	45.30%	0.568
Partial Activity	70.84%	0.461	68.67%	42.59%	0.557
Delayed Payment	69.67%	0.440	70.51%	40.14%	0.525
URSSAF debts	69.87%	0.443	69.31%	50.59%	0.526
Paydex	69.87%	0.442	69.05%	40.59%	0.526

TABLE 17 – Summary of Multi Layer Perceptron Metrics for each category over a 6 month period

MLP + dataset over a period of 12 months for a category	Balanced Accuracy	F_2-score	Precision	Recall	AUCPR
Financial Health	74.34%	0.538	86.28%	49.14%	0.691
Partial Activity	75.28%	0.555	84.64%	51.11%	0.693
Delayed Payment	74.74%	0.546	86.54%	49.95%	0.693
URSSAF debts	75.25%	0.555	85.12%	51.03%	0.690
Paydex	75.57%	0.560	84.26%	51.71%	0.694

TABLE 18 – Summary of Multi Layer Perceptron Metrics for each category over a 12 month period

A	Agriculture, sylviculture et pêche
B	Industries extractives
C	Industrie manufacturière
D	Production et distribution d'électricité, de g...
E	Production et distribution d'eau ; assainissem...
F	Construction
G	Commerce ; réparation d'automobiles et de moto...
H	Transports et entreposage
I	Hébergement et restauration
J	Information et communication
K	Activités financières et d'assurance
L	Activités immobilières
M	Activités spécialisées, scientifiques et techn...
N	Activités de services administratifs et de sou...
Q	Santé humaine et action sociale
R	Arts, spectacles et activités récréatives
S	Autres activités de services

FIGURE 8 – Correspondence between the sector code and the activity sector