Big data and its applications

Certificat Big Data

Professor: Georges Uzbelger – Spring Semester, 2017-2018

Thomas Bourany

Master Mathématiques de la Modélisation (LJLL) – UPMC – Sorbonne Université

1 Introduction and framework

The goal of this challenge is to predict if a transaction of a good purchased by a client through the "Price Minister" platform will be subject to a claim from the user (broken, not received, fake ...) or not. Given data and descriptive variables concerning the buyer (age, nb of purchases, location, etc), the seller (country, score, etc.) and the transaction (product type, price, shipping, etc.), one need to predict if there is a chance of a claim from the client. Results from this challenge will help to improve the quality of the service by Price Minister and the overall users satisfaction on its website.

This is a multi-class classification problem, evaluated through AUC weighted metric, which account for the ratio of true positive rate (sensibility of the classification) and the false positive rate (inversely related to its specificity) and the share of each class in the sample.

In this report, we describe how we process the data provided by Price Minister – which was a great part of the work for this project – and analyse the relative performance and the diverse classifications algorithm to handle and predict the share of claims in this dataset.

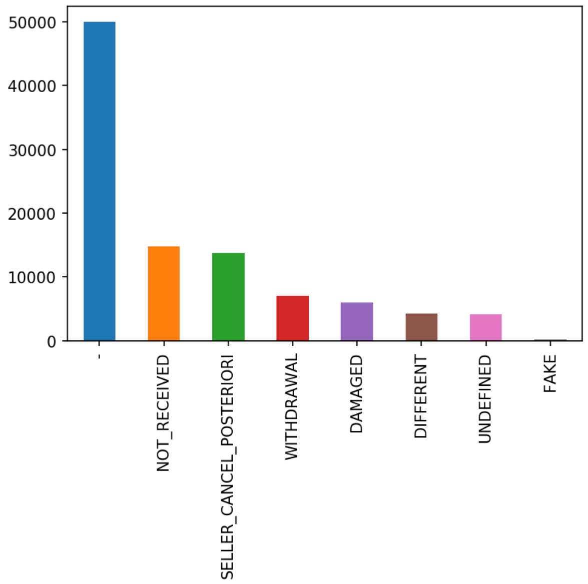
2 Processing the data

2.1 Description of the data

The data is composed of one output/dependent variable representing the case whether the transaction will lead to a claim, among the eight different following categories:

'OK', 'WITHDRAWAL', 'SELLER CANCEL POSTERIORI', 'NOT RECEIVED', 'DIFFERENT', 'UNDEFINED', 'DAMAGED', 'FAKE'

The *'OK'* or *'-'* label corresponds to the situation where no complaint has be issued by the client. The relative share of this target in the dataset is the following:



One of the main difficulty of this dataset is its unbalanced nature. The 'OK' label represent almost 50% of all observations and is four time more likely to happen than any other label – which makes sense given the business model of price minister as we will describe in the rest of this first part.

One way to deal with these data is to compare each label separately, by considering a "One vs. the Rest" procedure: considering eight 'new' output variables, each of them corresponds to one of the labels and is assigned the value $'1'$ when the observation has this label and $'0'$ when it is not. This procedure is the standard way to 'feed' multiclass output data to usual binary classification algorithms.

Concerning the independent variables, the list is given below and correspond to the diverse

** ID: identifier of the sample*

** SHIPPING MODE: mode of shipping of the product*

** SHIPPING PRICE: cost of shipping, if existing*

** WARRANTIES FLG: True if a warranty has been taken by the buyer*

** WARRANTIES PRICE: Price of warranty, if existing*

** CARD PAYEMENT: transactions paid by card*

** COUPON PAYEMENT: transactions paid with a discount coupon*

** RSP PAYEMENT: transactions paid with Rakuten Super Points*

** WALLET PAYMENT: transactions paid with PriceMinister-Rakuten wallet*

** PRICECLUB STATUS: status of the buyer*

** REGISTRATION DATE: year of registration of the buyer*

** PURCHASE COUNT: binarisation of buyer's previous purchases count*

** BUYER BIRTHDAY DATE: year of birth of the buyer*

** BUYER DEPARTMENT: department of the buyer*

** BUYING DATE: year and month of the purchase*

** SELLER SCORE COUNT: binarisation of the seller's previous sales count*

** SELLER SCORE AVERAGE: score of the seller on PriceMinister-Rakuten*

** SELLER COUNTRY: country of the seller*

** SELLER DEPARTMENT: department of the seller in France (-1 otherwise)*

** PRODUCT TYPE: type of the purchased product*

** PRODUCT FAMILY: family of the purchased product*

** ITEM PRICE: binarisation of the purchased product*

Strictly speaking, all these variables are coded by 'string' categorical values. However, at a first glance, the independent data can be divided among four groups:

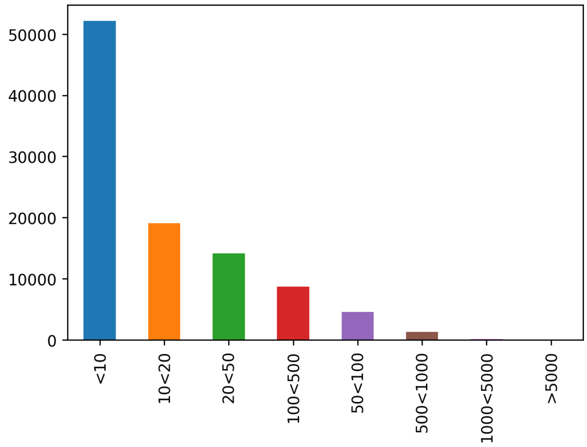
- Prices and potential 'numerical' variables (e.g. count and score)
- Type and family of goods (strict categorical data)
- Buyers and sellers locations
- Dates and time variables

The main idea of this part of the project is to find potential methods to process these data and to convert these values to numerical variables that can be treated by classification algorithms. These methods are described in the next sections.

2.2 Handling the prices and 'numerical' variables

Item prices, Shipping prices and warranty prices, and also purchase count and seller score count – whose repartition by category can be found in appendix – all display the same pattern as the one of the item price (obviously the most important variable of this dataset):

Figure 1: Item price of the transaction



The transactions with low prices are overwhelmingly more frequent than the transactions with high and higher price/cost. This first pattern provided the rationale for the procedure applied to the following variables, with more or less efficiency:

ITEM PRICE, SHIPPING PRICE, WARRANTIES PRICE, PURCHASE COUNT, SELLER SCORE COUNT

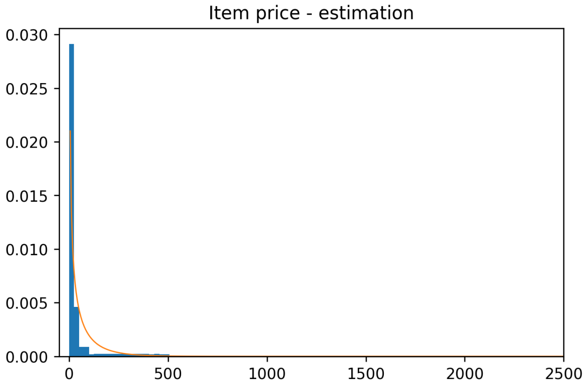
We will completely describe the procedure for the *ITEM PRICE*, but the method is exactly the same for these other variables.

When considering products with prices included in an interval such as between 10 and 20 euros, the very naive idea would be to assign the value in the middle of the interval, i.e. 15 euros. However, this would neglect the fact that the goods are very unequally distributed, especially for interval with large size (i.e. 1000<5000).

Using the obvious observation that a very large share of the sample has low values but that a non-negligible share of the sample still comprises extreme values, we approximate this distribution using a gamma distribution. The procedure is the following:

1. For the prices with J categories, consider the histogram values: n_j observations for the category between x_j and x_{j+1} (with $j \in \{1, \dots, J\}$)
2. Simulate a subsample of n_j observations uniformly distributed in $[x_j; x_{j+1}]$
3. Concatenate a sample with these $n = \sum_j n_j$ observations
4. Estimate the shape and scale parameters of a gamma distributions on this n-observations simulated sample (gamma.fit).
5. Compute the conditional expectation $\mu_j = \mathbb{E}(X|x_j < X < x_{j+1})$, for X a r.v. following a gamma distrib with parameters estimated in previous step [using IPP and numerical integration].
6. Assign the values μ_j for data in the category " $[x_j; x_{j+1}]$".

The estimation step can be displayed in the following graph.



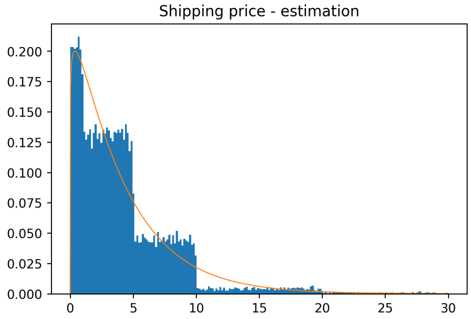
Numerically, to give a concrete example of the result, we obtain, for the item price the following values for each intervals:

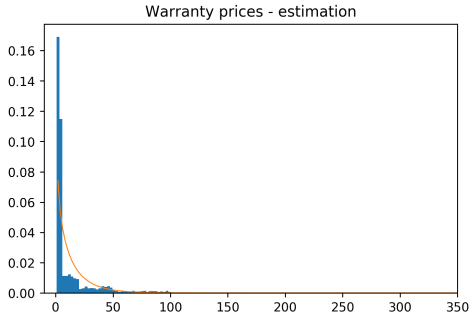
$$\{[0, 10]; [10, 20]; [20, 50]; [50, 100], [100, 500]; [500, 1000]; [1000, 5000]; [5000; 6000]\}$$

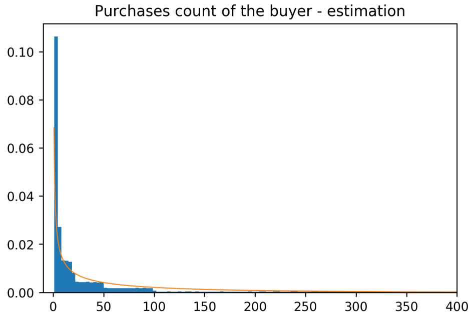
$$\{3; 15; 33; 72; 188; 607; 1213; 5026\}$$

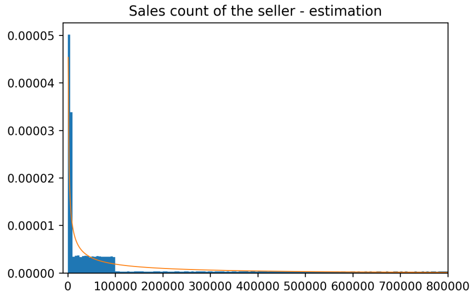
If one can display doubts about the result for medium range values concerning 'item prices', this method is much more precise than the naive method (we assign the value 188 instead of 300 or 1213 instead of 2500). This method is particularly effective to estimate the values of *shipping prices* or *values count and score* of buyers and sellers, as we see from the following graphs. However, the procedure for *item prices* or *warranty prices* seems slightly less appropriate, but still better than the naive method.

This procedure allows to obtain numerical where before one had categorical values. It allows a better understanding the pseudo numerical variables (i.e. included in intervals) than generating dummies variables (0 and 1) – a fate we keep for strict categorical variables (e.g. type and family of goods).



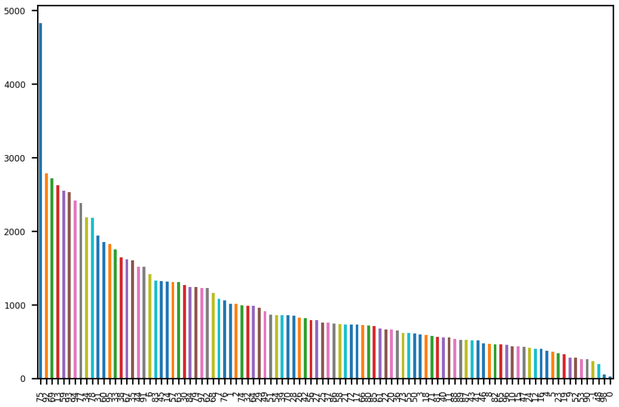


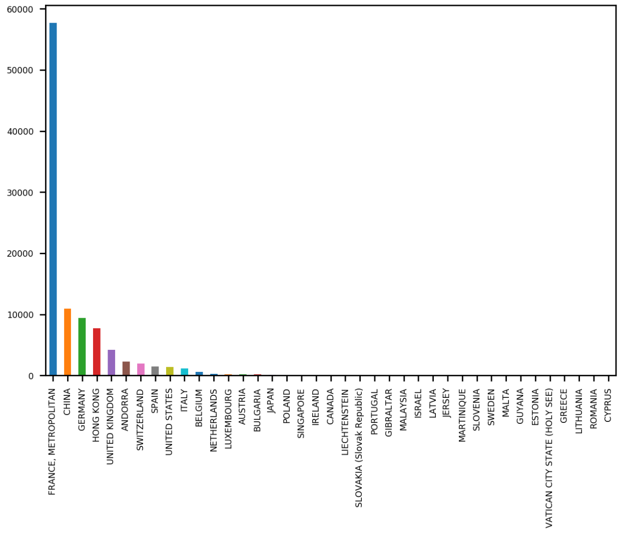




2.3 Handling the buyers and sellers locations – Distance matters!

Three variables of this dataset yields numerical variables that corresponds to departments of the location of the buyers, the department of the seller when this one is in France and the country of the seller when it is not the case. The two following graphs display the nature of the dataset concerning the first and third variables:





If the police is small, one can still observe that the first 4 départements of the buyers are the one of the 4 biggest cities/agglomerations in France (Paris, Hauts-de-Seine area, Lyon, Marseilles). For the country of the seller, one can see that beside France (and among its different départements, the same pattern for the most populated départements), the main countries are the world exporters (China, Germany, United States, Japan) and other countries surrounding France.

Considering the **Gravity model** of international trade – one of the most successful empirical model in economics for the last decades – and given the aim of this challenge to predict claim including *Not received* and *Damaged* issues, the main idea of this section would be to replace the buyer and seller départements by the ***Distance*** between these locations.

This is a well known fact in international economics that distance between importers and exporters can predict with high level of significancy the trade flows (as well as FDI flows and multinational corporation production implantation) between different locations in the world economy (much better than comparative advantage or factor allocations). According to us, this distance would be also successful in predicting claims issues in this dataset by PriceMinister-Rakuten.

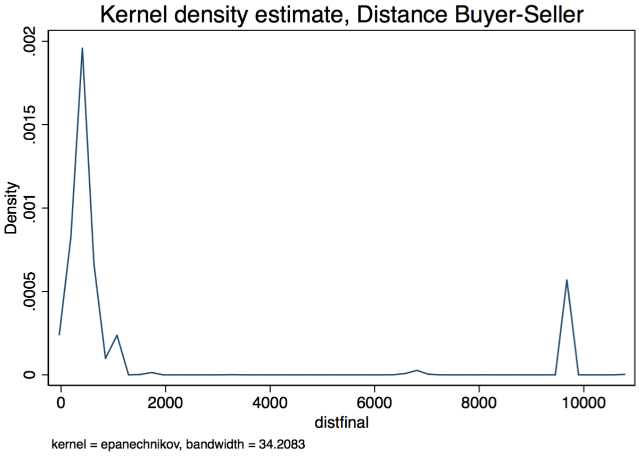
Using data for geographic distance – constructed by Head and Mayer (2002) and Mayer and Zignago (2011) – between countries for foreign location and constructing the database between the different location – using the procedure below – we replace the buyer and seller location by geodesic distances in km between them.

The geodesic distances are calculated following the great circle formula, which uses latitudes and longitudes of the most important cities/agglomerations in the country/départements,

The construction of the database between French départements, and between départements and neighboring location (Switzerland, Andorre, Luxembourg, Jersey) is the following:

- Import INSEE database matching department code and cities
- Import Geographical database (available online) matching cities and geographical (latitude and longitude) information
- Divide the dataset between each 'prefectures' of each French départements (buyer location) with seller location (départements or neighboring countries)
- Compute bilateral distance between each prefecture (between a and b), using the geodesic distances $dist = \arccos(\sin(\text{lat}_a)\sin(\text{lat}_b) + \cos(\text{lat}_a)\cos(\text{lat}_b)\cos(\text{long}_b - \text{long}_a))R_{earth}$
- Clean heavily the dataset, replacing outlier location (mainly départements that do not exist) in PriceMinister data by conventional values or dropping them.

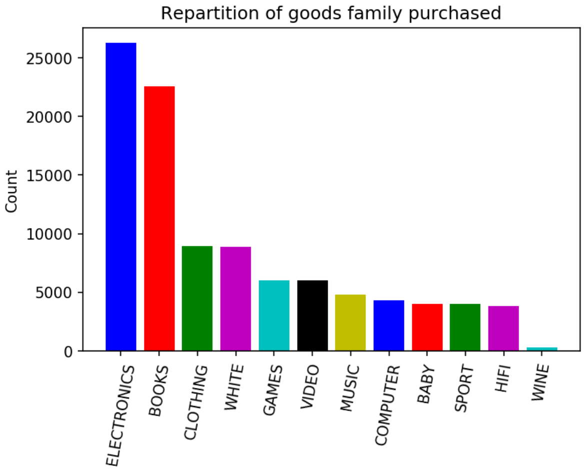
After this long (longer than expected) procedure, the result of the distribution of the distance is given in the following graph:



The spike at 9500 km represents the share of the transactions including French buyers with East Asian (Chinese, Hong-Kong, Japanese, Malaysian) seller. The biggest spike around 500 km corresponds to transaction inside France. For the seller variable, we also keep a dummy variable (0 for French sellers, 1 for Foreign seller). We also planned to include the population and GDP data per départements, but the cleaning procedure was too long and we plan to include these variables in an extension. To conclude, we are confident that this precise information will allow to predict efficiently the claim issue for this challenge.

2.4 Handling the type and family of goods

The processing of the family and type of product in the database is a standard procedure to deal with categorical data.



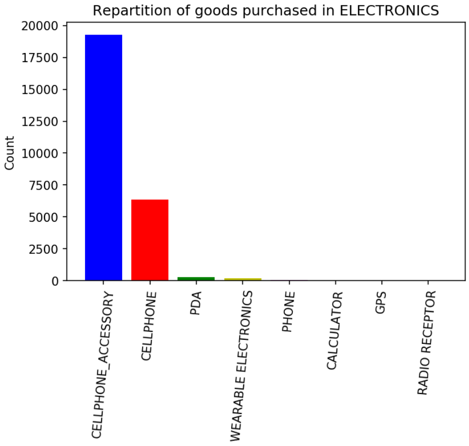
Here, we see that Electronics is the most frequent family of goods sold by PriceMinister, followed closely by Books. To handle these categorical data, the method is the following:

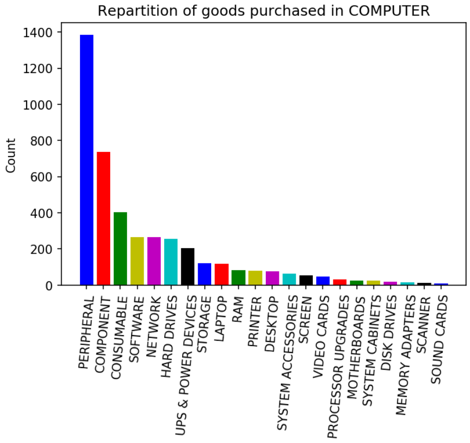
- Create K variables when the variable include K different categories
- Assign to the Kth variable a value 1 if the variable has the category K and 0 if not.
- Drop the variable corresponding to the most frequent outcome among the K categories (which is unambiguous, cf. comment below).

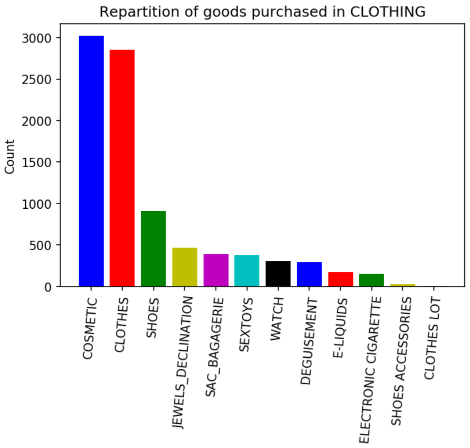
This procedure is also applied to the variables :

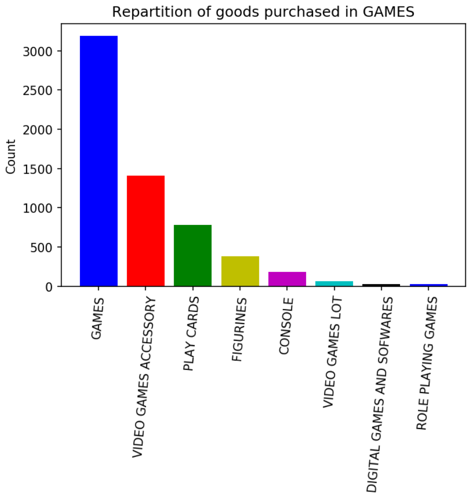
PRODUCT TYPE, PRODUCT FAMILY, PRICECLUB STATUS and SHIPPING MODE

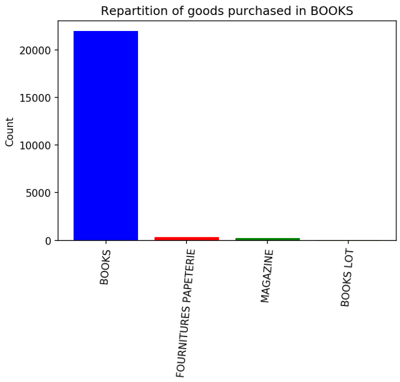
The main disadvantage of this procedure is to create a great number of dummy variables (one per category for each of these variables). An unattended consequence of this is to imply the ***'curse of dimensionality'*** since common statistical methods may not perform well in presence of a great number of dimensions (case for Likelihood estimations, regression models and KNN). In our case, we succeeded in keeping the number of dimension at ***'only' 167 variables*** (while binarizing each variables without the methods described above could have engendered hundreds of sparse variables).

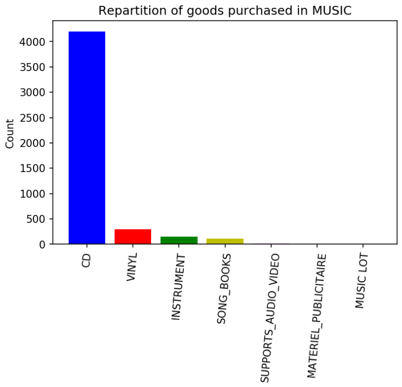


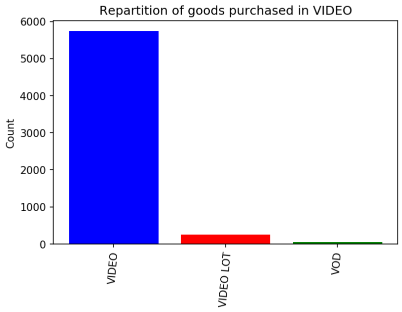












In the preceding graphs, we documented the category of *product type* variables. Again, we must emphasize the fact that the dataset is heavily unbalanced: many customers purchase the same category of goods. Two typical examples of transactions could be :

– A cellphone accessory (typically a smartphone protection) purchased at a low cost (less than 10 euro, cf. estimation above), from China or from retailers in France.

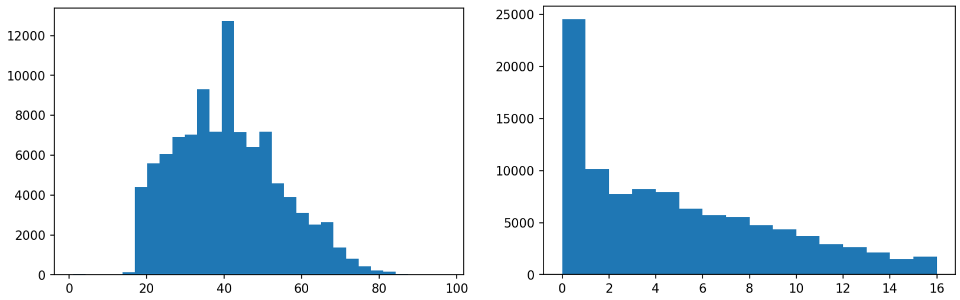
– A book, a DVD or a CD (again at a low price) from French editors or French retailers.

2.5 Other variables – age and registration date

Two other variables correspond to easily convertible data. The birth date, the date of registration of the client and the date of the transaction can be easily converted to a numerical value. Subtracting from the last data known (December 2017) these different dates, we obtain the following numerical variable with repartition:

If the age of the buyer is somehow conventional – beside the spike at 42, since we replaced the missing and outliers values (age >110 y) by the mean of the sample, the time from registration again displays an unbalanced shape: most costumers only registered to make their first and only transaction.

Figure 2: Buyer age (left) and Time from registration (right) in years



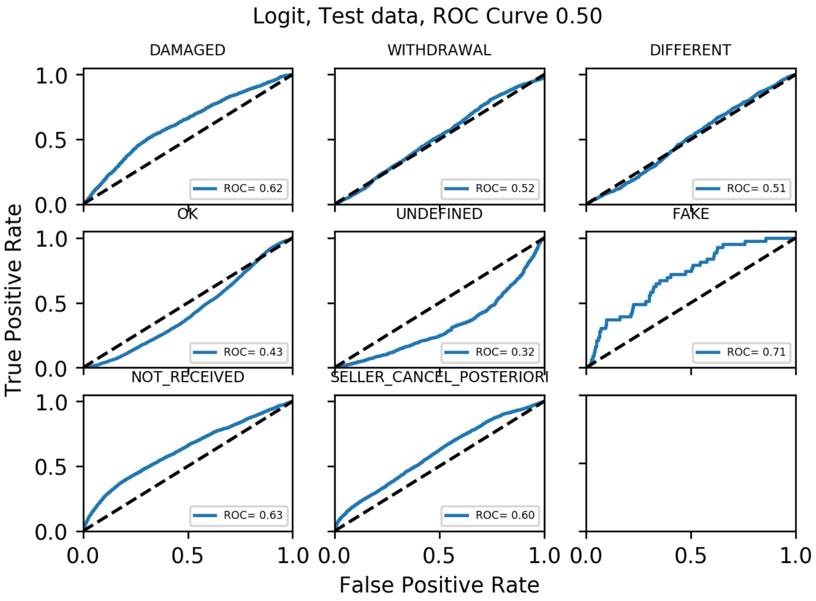
3 Classification algorithms

3.1 Simple Logistic regression

Using Sklearn, we perform a simple multiclass Logistic regression, using all the numerical variables (either dummy variables or converted data obtained using the procedure described above) of the data, we obtain a rather poor performance. The Roc Auc is only of 50%. The accuracy is also very low at around 49%.

This number emphasizes the fact that Logistic regression is a poor classifier to discriminate the true positive and the false positive, and could assign/predict to most of the observations the value ’OK’. We will see that such pattern is also common for deeper Neural networks (from few layers up to 10 layers, both with large or narrow (in terms of numbers of neurons) layers), showing that these two types of methods may not be appropriate to deal with this classification.

However, when displaying the ROC AUC for each label, we can observe the following result:

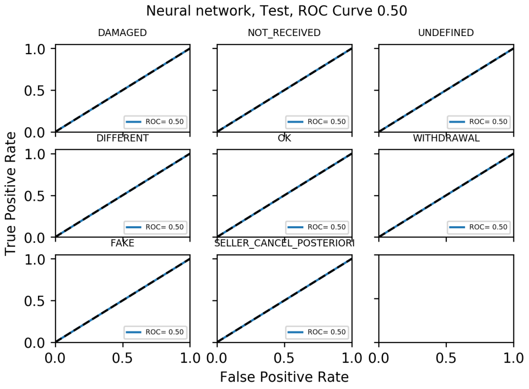


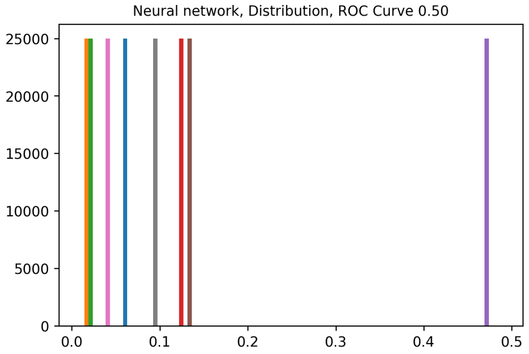
The main conclusion we can draw from this graph is the fact that several class are better identified than others : Fake, Not Received and Damaged label are well described compared to the rest, with ROC values above 60%. In particular, we could say that the distance variable we constructed could have played a role in this identification, since counterfeit and damaged products are often sold by retailers from far foreign countries.

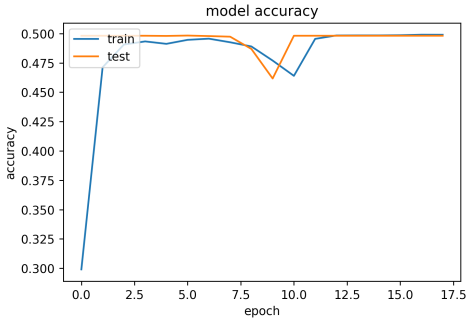
In comparison, the 'OK' label have a high rate of false positive rate, implying that the algorithm does not discriminate and assign the value 'OK' to many observation that could have had other labels (withdrawal, different or undefined).

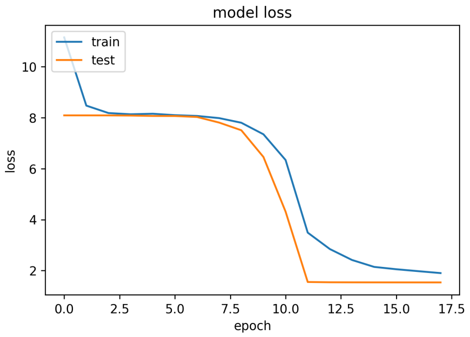
3.1.1 Neural networks

The Neural networks, without further pre-processing of the data have also poor performances, yielding accuracy and Roc Auc rates of around 0.5. The displayed curves display the same patterns and robustness checks, changing the hyperparameters – number of neurons, number of layers, depth of the network, size of the batch – always results in the same non-discriminative behavior. The accuracy and the Roc Auc rates stay constant at 50% after a certain threshold where the predictions of the model stay constant: the algorithm will classify *all* the observation with the same value, corresponding to a constant probability of the class in the sample (displayed on the first RHS graph below). The experiment below is just an example with 4 intermediate layers (respectively 50, 30, 15 and 8 neurons and relu activation), 18 epochs, 200 batch size, and cross-entropy loss. The same pattern shows for alternative specifications.







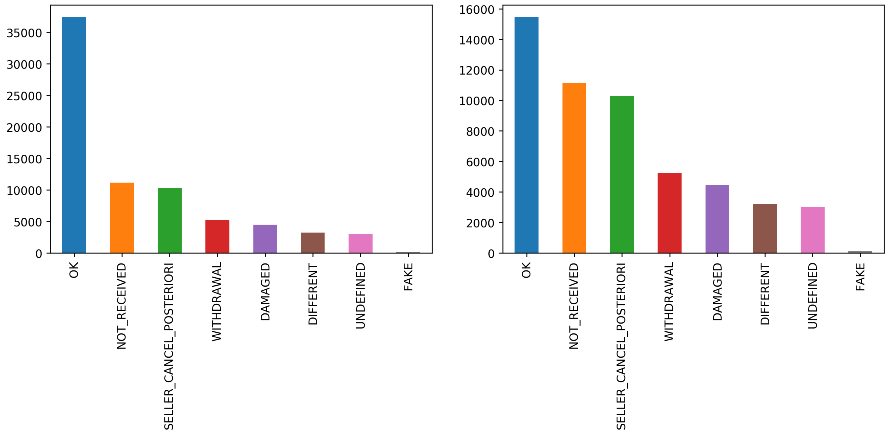


3.2 Rebalancing the dataset

Considering the poor performance of these two algorithms, we consider rebalancing the training dataset. In the training dataset, comprising 70% of all observations we had half of observations being the 'OK' class. One simple idea to rebalance the dataset is to train the algorithms using only a subset of the 'OK'-class observations.

In order to implement this idea, we reduce the dataset, dropping randomly 50% of the 'OK'-label observations. This corresponds to 20'000 obs. for a training dataset of 75'000 obs. The result in terms of class repartition is displayed in the following graph.

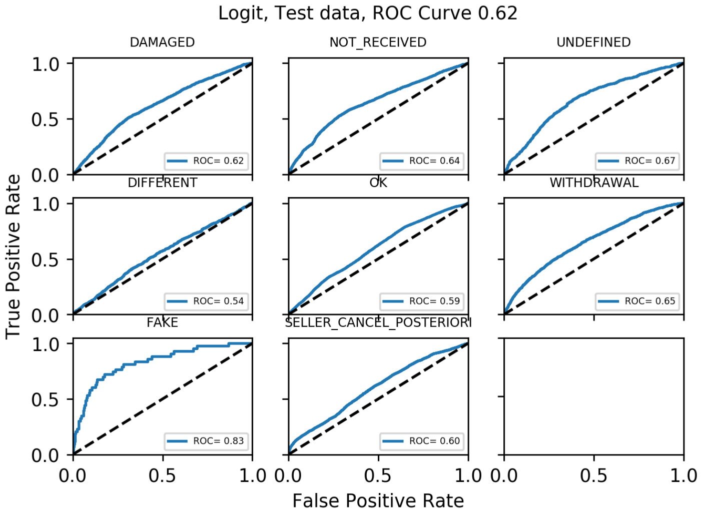
Figure 3: Class repartition - Unbalanced (LHS), Rebalanced (RHS)



3.3 Logit and Neural networks using the rebalanced dataset

In the following, we only display the result for *test* data, whereas training of Logit and Neural Networks algorithms was performed on the rebalanced (training) subsample.

For Logistic regression, the performance is the following, in terms of ROC AUC. Obtaining a 60% ROC AUC for *test data*, one must still underline the fact that the accuracy (share of the sum of true positive and true negative rate) is reduced to 47%. This is again a typical example of the 'accuracy paradox': for a minor reduction in accuracy, the algorithm performs much better in terms of roc auc. In particular, the discrimination power for the 'OK' is much better, with a false positive rate much below.



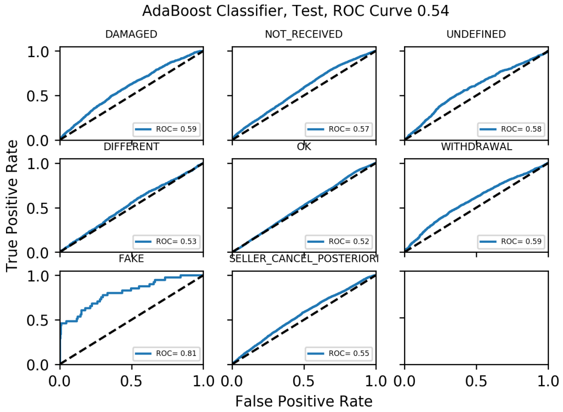
Unfortunately, the Neural networks trained on this rebalanced dataset do not display better performance to predict the claim issue. This may imply that Neural Networks are not adapted to such problems. It is commonly known that unbalanced dataset with categorical variables are better handled by decisions trees. Therefore, such ensemble methods are the ones we consider for the rest of this section.

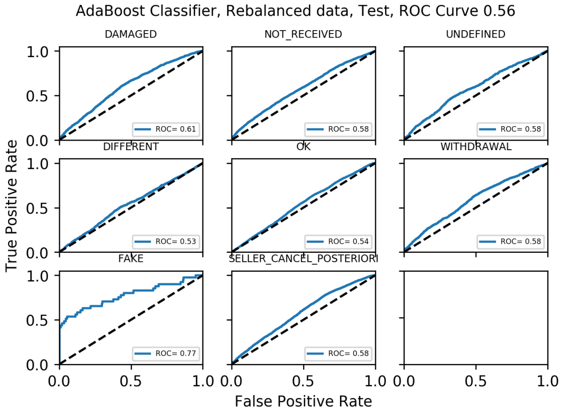
3.4 Gradient boosting – Adaboost

The gradient boosting algorithm is a linear combination of weak learner, typically a set of simple decision trees, built to classify data in groups according to their situation with respect to a threshold. This linear combination being a complex problem – optimization problem over a large space making it a greedy algorithm – the solution is constructed recursively, simplifying the task of finding an additional weak classifier:

- Take the classifier being a linear combinaison of given weak learners.
- Compute the gradient of the loss for this function.
- Select an optimal (simple) weak learner close to this gradient (square minimization)
- Select the optimal step size
- Add the step time the weak learner to the linear combinaison, returning this boosted classifier

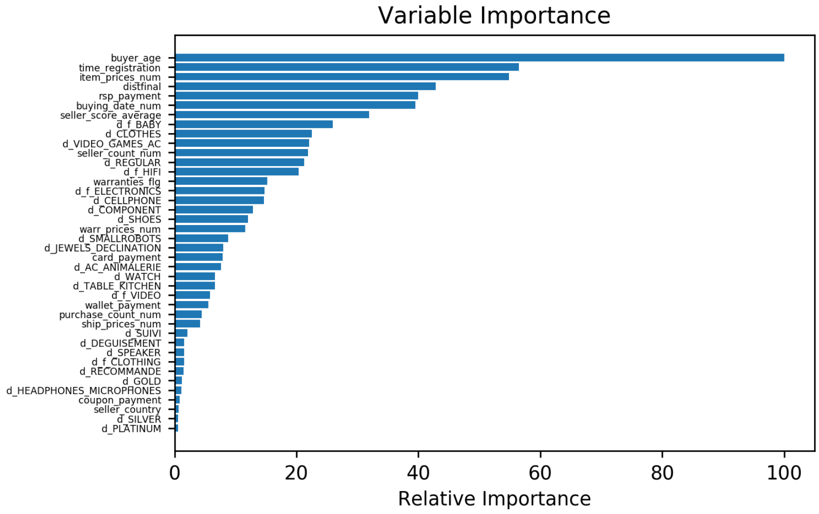
Adaboost corresponds to this algorithm when considering the exponential loss. It increases the loss for misspecified observations. The performance of this method is displayed in the following graphs. With different specification, we can observe that the overall efficiency in terms of Roc Auc stay stable around 55%. In particular, the predictive power for the 'OK' is relatively low, with a significant number of false positive prediction. However, the RHS graph shows that the performance is slightly better for this class when training the data set on rebalanced data.





The next interesting characteristic of classification trees, is that we can understand the importance of a feature for the classification algorithm. The relative rank (i.e. depth) of a feature is estimated by the expected fraction of the samples they contribute to. Features used at the top of the tree contribute to the final prediction decision of a larger fraction of the input samples. The result for the decisions trees used as weak classifiers in Adaboost is displayed in the following graph.

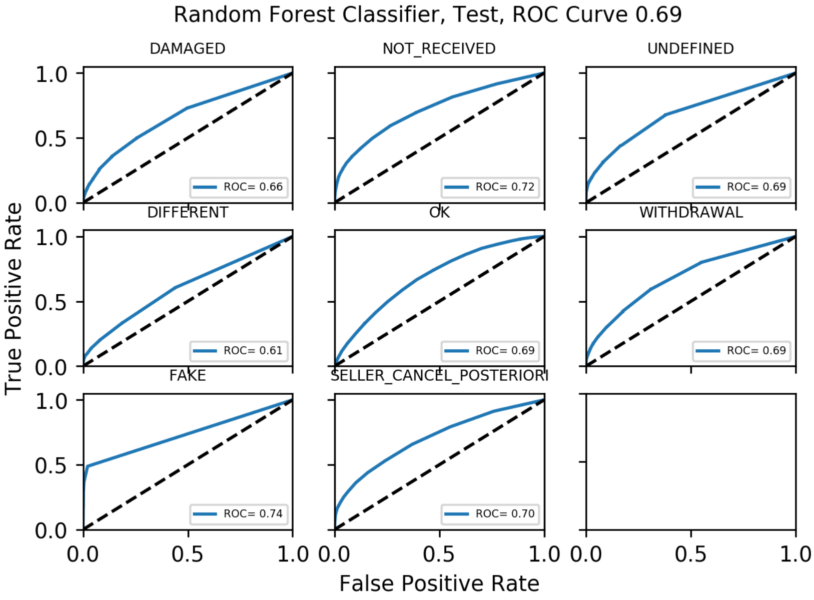
We can observe that the age of the buyer is the first and most important variable (probably at the root of the trees), followed by the length of the time since registration on PriceMinister platform. The forth variable (which is second or third in different specification), is the distance between the seller and the buyer. As explained by our intuitions in the first sections, this variable might explain well the probability of damage or 'not received' motive for a claim.



3.5 Random forest

In this classification challenge, it may seem intuitive that decision trees can be efficient: Above a certain threshold of a given variable, say the item price or the time since the registration in PriceMinister, the client might be more likely to complain (or not!). The poor performance of Adaboost is disappointing and we now turn toward Random forest algorithm.

The result is displayed on the following graph, where the performance is systematically better for all the classes and the overall Roc Auc. The Accuracy, however, is not better than simple logistic regression.



4 Assessment and conclusion

To predict if a transaction of a good purchased by a client through the "Price Minister" platform will be subject to a claim from the user. We perform multi-class classification algorithms, evaluated through Receiver operating characteristic. The relative performance of these algo is displayed in the following table.

Table 1: Assessment of the different methods

Algo	Specification	Accuracy	*Roc Auc per class*								*Weighted*
			Dam	Diff	Fake	Not Rec	OK	Cancel	Undef	Withdraw	*Roc Auc*
Logit	Train data	0.497	0.625	0.563	0.848	0.641	0.611	0.603	0.664	0.653	0.626
Logit	Test data	0.504	0.631	0.578	0.837	0.627	0.605	0.603	0.633	0.664	0.621
Logit	Rebalanced train	0.315	0.604	0.520	0.829	0.639	0.582	0.616	0.659	0.638	0.617
Logit	Rebalanced test	0.466	0.633	0.539	0.839	0.631	0.580	0.599	0.651	0.663	0.615
Neural network	4 layers, 20 epochs	0.146	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500
Adaboost	200 estimators	0.325	0.590	0.527	0.808	0.566	0.519	0.554	0.580	0.586	0.543
Adaboost	200 est., rebal. train	0.263	0.605	0.531	0.774	0.577	0.537	0.577	0.580	0.585	0.557
Random forest	20 estimators	0.361	0.664	0.607	0.736	0.719	0.688	0.704	0.690	0.688	0.690