PINEAU Edouard
SAVARY Adrien

# DATA CHALLENGE

Goal of the challenge: predicting which pipes have the highest failure risk in respectively 2014 and 2015. The objective for Veolia is the optimization of the maintenance procedure on the pipeline network.

## INTRODUCTION:

The main difficulty of this challenge is to deal with a very imbalanced dataset (~1% minority class). We have developed many different approach during the year but we present here the most significant ones and obviously ones that lead to our best scores.

The score of the challenge is computed using a weighted mean of the AUC of each year prediction with the formula:

$$0.6\times AUC(2014) + 0.4\times AUC(2015)$$

The **sklearn** library **metrics** propose a function **roc_auc_score** that we used for cross-validating our choices.

## I. Data analysis and pre-processing:

The matrix of data supplied is composed of three categorical and four quantitative features. The three categorical variables are anonymous as well as one quantitative. Two others are time-related (**YearOfConstruction** and **YearLastFailureObserved**) and the last one is the Length of the pipelines.

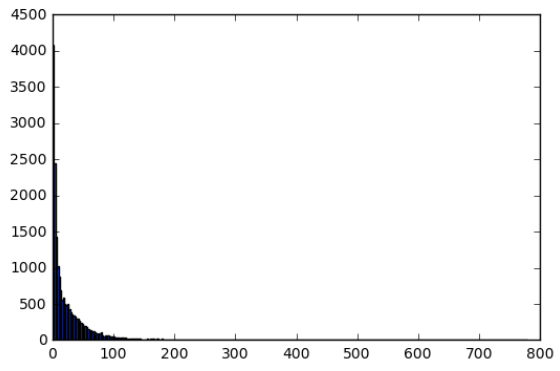
The main objective of the pre-processing would be to make the variables the as meaningful as possible. We first transform and select from variables we dispose the one that are not clean.

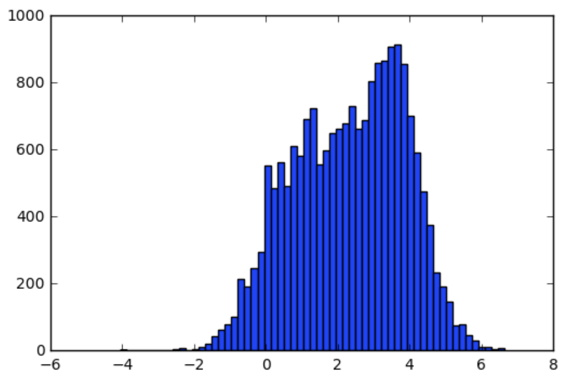
- The script categorical data will be transformed naturally into quantitative categorical data in order to be read by sklearn functions. A first approach using a dumb quantification with the construction of orthogonal latent variable caring zeros and ones representing each category of each feature was considered as a terrible choice after many tests. We finally chose to use the **LabelEncoder** function from **preprocessing** library that replaces the categories by integers within each variable. This construction proved its efficiency in the results

- From the feature **YearLastFailureObserved**, we extract the information of the time to failure of a pipeline and create a variable **TimeToFailure**, with 0 where no failure occurred. This variable has not particular objective of being used for prediction, but care information that we might require later

- The feature **YearLastFailureObserved** will be replaced by a categorical variable with two categories representing the occurrence or not of a past failure. This variable has not particular objective of being used for prediction, but care information that we might require later

- We decided to transform the quantitative data in order to make it as Gaussian as possible. The goal was to find back a most simple and classic shape for our data that might simplify our approach of clustering and outlier detection. For example for the **Length** variable, we applied a log and obtained:


before transformation


after transformation

We then rescaled the dataset using function **scale** of the preprocessing **library**. We then obtain clean dataset to ease the study.

Observing the data, we determined that we would face two main troubles:

- The very low-dimensionality of the data might make the distinction between good and bad pipes very difficult
- Looking at the representation of the continuous features in the 3-dimensional space (Length, AgeOfPipeline, Feature3), the anomalies are very embedded within the all cloud and the distinction from those features might be tricky

We decided to use different prediction methods and to propose a final score with an aggregation of the results. To tackle cleverly the problem, without testing the all panel of methods proposed by scikit-learn, we based our first tries over the sentences of Chandola and alumni (2009):

*Normal points occur in dense regions, while anomalies occur in sparse regions*

*Normal point is close to its neighbours and anomaly is far from its neighbours*

## II. Prediction methods:

We first began to look at examples of anomaly detection in order to have a main idea of the troubles encountered when dealing with unbalanced data: fraud detection, cancer prediction, stellar anomalies… Predominant methods got out according to the type of data we needed to treat:

- For continuous data: probabilistic, distance-based, domain-based methods
- For categorical data: tree methods

The first point is based on the idea that normal observations are dense, the second elaborate the idea of logical clustering that discriminate unusual observations.

1. <u>Density estimation</u>:

The objective is to evaluate the density of the data and find a threshold to discriminate highly unlikely points. Those points would be anomalies within the dataset. We tried:

- Parametric approach: GMM density estimation using **GaussianMixture** from **mixture** library

- Nonparametric approach: Kernel density estimation using **KernelDensity** from **neighbors** library

We then selected for each the 10% less likely points, and found an empty intersection between failed pipes and predicted failures. We rapidly concluded that probabilistic approach might be inappropriate for our problem.

2. <u>Distance-based/clustering-based methods</u>:

- <u>K-nearest neighbours</u>:

A natural distance-based method for anomaly detection is the nearest-neighbour approach. It is based on the assumption that normal data points have close neighbours in the normal training set while novel points are located far from those points. A point is declared outlier if located far from its neighbours. But as we saw before the high level of similitude between failed and normal pipes makes a spatial discrimination with KNN very cheap.

A test with the function **KNeighborsClassifier** from **sklearn.neighbors** confirmed the impossibility to discriminate by spatial proximity. We see that thinking to micro does not work with our problem.

Alternative methods we tried were semi-supervised clustering procedures: the point is to clusters of study:

- <u>K-means method</u>

We built a function `which_closer` that takes a point and a list of centres and say from which member of the list the point is the closest.

Then using the function **Kmeans** from **sklearn.cluster** library I take the few examples of failed pipelines that we get from the dataset (from here comes the semi-supervision) for each year 2014 and 2015 and find clustering centres. We reiterate for the normal examples of the training set and find centres for each year.

Once we had the centres, in order to return a score vector, we put zeros when the closest centre was of normal type and the inverse of the distance in the continuous feature space for others. This way, we highlighted the predicted failures without being too hard at discriminating the potentially failing pipes.

- Gaussian Mixture ellipsoid clustering:

We already tempted the density estimation with threshold for the detection of probabilistic anomalies, but did not exploit de geometrical tool brought by the GMM: ellipses.

We created the classes **GmmAttributes** and **Ellipse** for the construction of the method. The principle is to find the ellipses that come from a Gaussian mixture modelling calibrated over the respectively "failed pipeline" group and the normal group and to determine a score considering the belongings. The more a point is related to ellipses of the group "failed pipelines", the higher the score will be. This way, we can use the spatial similarity without building a hard classifier. It is more a question of average behaviour.

We suffered two main problems:

- When the number of clusters grows, the convergence of the GMM is not robust, even with Bayesian selection of the optimal number of component for the mixture. Then we are limited to gross large ellipses
- Random level of over fitting according to the set over which the model is calibrated

3. Domain-based methods:

- SVM one-class:

A classical approach of novelty detection is the consideration of only one class because of the highly unbalanced dataset we have.

Intuitively for our problem, considering the high similitude of the different classes over the continuous feature space, the opportunity to increase the dimensionality of our space by using kernel trick might be salutary.

Despite this ability to distort and over-dimension our space, the method is very unstable and we find back the troubles we encountered with any spatial approach of the problem. The high level of imbrication of failed pipes within the normal ones makes one-class SVM inappropriate for our problem.

- Isolation forest:

This unsupervised nonparametric procedure for multivariate data anomaly detection that focuses on continuous-valued data only. It measures the susceptibility of an individual to be isolated from the rest of the set. We do not expect this algorithm to be accurate but only to create an under-sample of our set that presents a higher proportion of suspiciously behaving individual.

The tests over our sub-samples demonstrate the good property of completing some detection unseen by other methods. The fact that we use an aggregated score enables then to use it to confirm or complete what we saw with distance based-method.

Nevertheless, the robustness problem still holds, linked to important over fitting of the train subsample.

4. Tree methods

Very used and known procedures, for example in credit default prediction for financial institutions, are the random forests and extra-tree classifiers. Those methods facilitate the treatment of unbalanced datasets, in particular for categorical features.

Unlike the previously presented methods, this one does not suffer robustness problems. No matter the under-samples selected, the cross-validation score stays relatively constant.

Moreover, it has an equivalent ability as isolation forests: it treats some part of the feature space unseen by the distance-based methods.

5. Adaboost (with SMOTE):

The main idea of Adaboost is to train multiple classifiers (trees), every tree is computed using information of the previous tree. Taking a closer look at the algorithm we see that every example is weighted according to its difficulty with the normal classifier. Finally, the final classifier is obtained with a weighted average of all trees. In our case, this method looked very promising because of its capacity to deal with outliers.
This algorithm is working very well with the SMOTE oversampling method (described in part III) as we get more "difficult" points to put weight on. When we write this report, it is the method that gave us the best results so far in the test set. (~87%)

## III. Augmentation tools and results:

1. Smote: Oversampling Methods to deal with an imbalanced Dataset

The idea of Smote is to create synthetics samples of the minority class in order to introduce diversity in the training set and fix the imbalanced property. The idea developed in the [3] is the following: we consider a vector of the minority class, and then we find its nearest neighbour and compute the difference between the two. Then we just multiply it by a number between 0 and 1 and add it to the original vector. Like this, we created a new data point along the data set. The synthetic examples cause the classifier to create larger and less specific decision regions: it reduces overfitting which is one the major issue when working on imbalanced data sets. The strategy of just duplicating samples is limited because doesn't prevent overfitting.

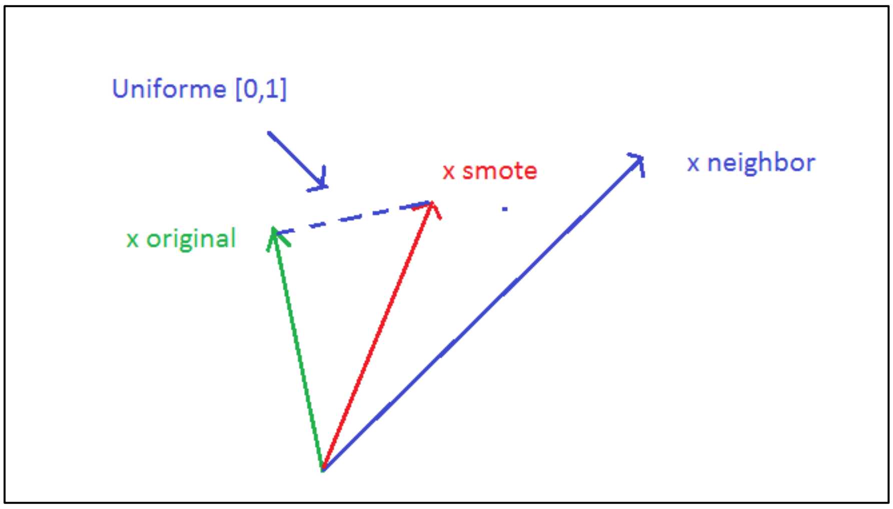


Figure: Smote intuition

2. Results of the methods, empirical advantage of aggregation:

Using an aggregation of the algorithms described above, with different treatment for the years 2014 and 2015, we achieved the score of 0.89 of AUC. The utilization of SMOTE was the key to achieve this level, a ceiling at 0.83 existed with the imabalanced dataset, only using GMM, Kmeans and random forests since other methods did not detect any positive example because of the imbalanced calibration. The AUC score enables the aggregation of results, with no necessity to send back 0 and 1 vector, but scores that give the intensity of chance to be a failing pipeline.

## Conclusion:

The most difficult part of the challenge was to identify ways to deal with the 'imbalanced' feature of the data set. Small mistakes slowed us down but we tried to come up with simple ideas and improve slowly but surely our score. We went from a "made by hand" model to a more sophisticated Gaussian mixture averaged with some tree methods to finally get the best result with boosting methods using Smote. The most important part of our work is not represented by this last result but more by all the steps we put together to understand the problem. A way to get better result would be to use ensemble methods to get the best of every classifier and prevent overfitting.

## References

[1] Liu and Ting (???), Isolation-based anomaly detection

[2] Pimentel, Clifton, Tarassenko (2012), A review of novelty detection

[3] Chawla, Bowyer, Hall, Kegelmeyer (2002) SMOTE: Synthetic Minority Over-sampling Technique