Data-Driven approaches in Anomaly Detection

Anass Akrim
Ecole des Mines de Saint-Etienne
Saint-Etienne, France
Email: anass.akrim@isae-supaero.fr

Mihaela Juganaru-Mathieu
Institut Henri Fayol
Ecole des Mines de Saint-Etienne
Saint-Etienne, France
Email: mathieu@emse.fr

***Abstract*—The fight against frauds became a major issue with the development of organizations. Companies are currently operating in an uncertain environment and may face different challenges : combating insurance fraud, credit card frauds...**

Therefore, it becomes necessary to equip itself with the tools to face this threat and find a solution to detect, manage frauds.

This paper proposes a novel deep learning-based approach (Restricted Boltzmann Machines) for anomaly detection. We compare the effectiveness (precision and recall) of this semi-supervised method with three main unsupervised statistical data-driven algorithms used for anomaly detection : DBSCAN [], OCSVM and Isolation Forest.

Keywords: neural networks, financial Fraud detection, unsupervised, semi-supervised, machine learning, deep learning, real-time detection.

I. Introduction

According to PwC's "Global Economic Crime Survey 2018", almost three-quarters of the French businesses report having been victims of fraud in the last two years, and about 12% of the French businesses surveyed reported having been victims of fraud with a loss of more than 1 million.

Therefore, it becomes necessary to find a solution to detect and manage frauds quickly.

Today, With the development of AI and the volume of data growing exponentially, we see the possibility of automating fraud detection and reducing losses.

Recent papers proposed the use of RBM to detect frauds. Indeed, according to [UFAA13], RBM allows to learn high level features from the training data set, and can distinguish features/patterns of non fraudulent observations, which allows us identify outliers .

A Restricted Boltzmann Machines (RBM) is able to detect outliers in real-time. Trained only on normal (genuine) transactions set, the model will learn the pattern of normal transactions, can reconstruct what non fraudulent transactions looks like and learns how to discriminate whether or not new transactions belong to that same class.

According to [Res18], the data is first splitted in the ratio 80:20 , and the model is trained on the training set. It is necessary to train the model on a data set with non fraudulent transactions, so we can identify outliers. The whole process looks like this :

After convergence, we compute the 'free energy' function, or score function used is this approach : the higher the free energy, the higher the chance of a transaction x being a fraud.

A good tuning RBM parameters tutorial can be found in this paper [Hin10].

According to [Res18], only one iteration in the learning process is sufficient and gives good results, and with good parameters : an AUC score of 0.96, which means that 96% of the predictions are correct and the model fots for a large data set (284,807 transactions with only 492 frauds).

Fraud detection often comes down to outliers detection, in which a dataset collected is scanned to find potential anomalies in the data. Previously, this was done by employees who checked all transactions manually. With the growth of machine learning, artificial intelligence, deep learning and other relevant areas of information technology, it becomes possible to automate this process and to save some of the intensive work devoted to fraud detection, using specific techniques from Machine Learning and Deep Learning.//

Scientifically, fraud detection could be considered as a supervised classification (at best), or unsupervised in general. We talk about unsupervised learning when no training data set is available nor needed. An outlier, or anomaly, is defined as an "isolated case", or deviant. It's something with features deviating from the norm.

But the volume of frauds is much smaller than the volume of correct transactions.

For this reason, fraudulent transactions can be considered as outliers and specific semi-supervised methods are still under consideration.

II. Methodology

A. *Problem statement*

In statistics, an outlier is an observation that so much lives from other observations that it tends to arouse suspicion. In other words, an anomaly is an observation which differs from an overall pattern on a sample. An observation is described as an anomaly when it has one of the following characteristics :

- Statistical properties different from the rest.
- Lies in a region with low density.
- In modelling, an outlier is an an observation that is set apart from the others.

We can find outliers in two types of data set :

The Machine Learning techniques are the object from now on of a considerable attention on behalf of the researchers in detection of anomalies to remedy the weaknesses of the fraud detection techniques used up to here. The detection of

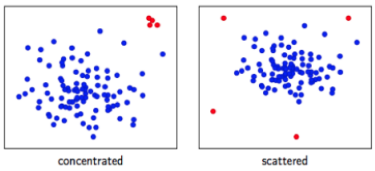


Fig. 1. Outliers set apart from others (concentrated), outliers in a region with low density (scattered).

anomalies can help effectively to detect frauds, discovering a strange activity in large data sets.

The detection of the absurd values or the abnormal transactions is the problem of detection in a database, which are very different of what looks like a 'classic' transaction in the whole of data that a financial institution has.

In Machine Learning we use clustering techniques, or unsupervised approach, to identify anomalies, methods that do not require training datasets and are based on two basic assumptions:

- First, they assume that most transactions are not fraudulent and that only a small percentage is abnormal.

- Secondly, they assume that the set of fraudulent transactions is statistically different from the set of non-fraudulent transactions.

Generally, the unsupervised Statistical algorithms use distances or densities to estimate what is normal and what is aberrant.

We can list three categories of unsupervised Statistical Learning algorithms:

- Clustering by density (DBSCAN, OCSVM);
- Clustering by 'isolation' (Isolation Forest).

However, these three algorithms

These techniques are efficient in detecting anomalies, but on a small data set (sources !!!) and require a lot of memory and time for computation.

Nowadays, due to the large number of transactions per second, we have large volume of data. And we are exploring the use of deep learning approaches to tackle this problem.

Deep learning (also known as deep structured learning or hierarchical learning) is part of a broader family of machine learning methods based on learning data representations, as opposed to task-specific algorithms.

Recent papers proposed using neural networks for fraud detection, especially for financial frauds, which implies large data sets. We are talking about a semi-supervised Deep Learning method : Restricted Boltzmann Machines.

It is an algorithm which is useful for dimensionality reduction, classification, regression, collaborative filtering, feature learning, and topic modeling. RBM shares a similar idea as Autoencoders [], but it uses stochastic units with particular distribution instead of deterministic distribution.

The motivation for this work is to explore and develop the framework or Restricted Boltzmann Machines (RBM) to set a robust method capable of identifying financial fraud in real-time, or anomaly in a data set collected, through semi-automatic learning.

In this study, we are looking to set a robust methodology, while calibrating and optimizing the predictive model built, and compare the RBM method with the statistical methods used.

- Very large data set ()

-The data we process are unlabelled : this makes it difficult to validate the performance of each machine learning model used for fraud detection.

B. Approach :

In this study, we are looking to put in place a robust method capable of identifying financial fraud in real-time, or anomaly in a data set collected, through semi-automatic learning :

- Set a robust methodology, while calibrating and optimizing the predictive model built, with Restricted Boltzmann Machines.
- Comparison of the methods used and showing novel approaches for detecting anomalies.

III. Algorithms

A. DBSCAN (density-based spatial clustering of applications with noise

DBSCAN is a relatively recent clustering algorithm (Ester et al. 1996), based on density clustering, capable of discovering clusters of any size or shape and accurately identifying outliers.

Unlike other clustering methods (including K-means) that assign each point to a cluster, DBSCAN is able to recognize the outliers and not group them together.

The DBSCAN algorithm requires two parameters:

- *Epsilon*: maximum distance between two observations (the distance used is the Euclidean distance in this study) to be considered as part of a group (or cluster);
- *MinPts*: minimum number of observations to be able to form a cluster.

Indeed, instead of requiring a number of clusters (like K-means), DBSCAN may be able to automatically compute any number of clusters, based on given parameters.

We can see in the Figure above that:

- Depending on the value of the MinPts parameter, Cluster 2 and Cluster 3 can be divided and considered anomalies.

- Depending on the value of the Epsilon parameter, Cluster 1 could be divided into several clusters (and outliers) if we use a smaller epsilon; or could be merged with Cluster 3 if we use a larger epsilon.

However, these parameters are not known in advance, and they should be selected carefully. This therefore requires a rigorous study of the data set available. This corresponds to one of the difficulties of the DBSCAN algorithm: simple to use, difficult to set exactly.

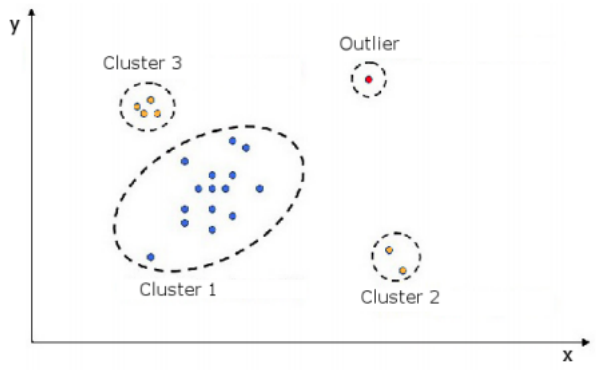


Fig. 2. Possible clustering using DBSCAN algorithm.

B. *OCSVM (One-Class Support Vector Machine*

OCSVM is an extension of the original SVM algorithm.

It is an approach that seeks to put in place an “envelope”, or a support for “normal” observations, defined by one class classification separators (SVM) (Schölkopf et al. 1999).

The principle is to pose an optimization problem with the objective of separating the data, maximizing the margin according to the parameters of our problem.

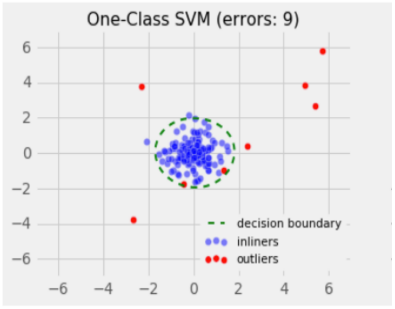


Fig. 3. Possible clustering with OCSVM algorithm.

The solution produces a binary function that is worth +1 in the smallest region capturing the data, inliers and -1 elsewhere, the outliers.

The One-Class SVM requires two parameters:

- ν: Either the upper limit of the number of outliers expected.

- γ : in theory, the coefficient of the kernel function of the algorithm, which determines the separation boundary between the two classes and therefore the number of outliers.

As for the classic SVM, this algorithm is based on the use of a kernel (linear, polynomial, sigmoid or RBF) and uses the kernel trick to calculate the scalar product between data points represented in a high-dimensional space to find a hyperplan separator.

C. *Isolation Forest*

Introduced recently, the algorithm ’Isolation Forest’ [LTZ08] is based on the same principle as that of ’Random Forests’, but is based on the construction of a completely random set of trees: isolation tree. This algorithm uses neither distance measurement nor density measurement to detect anomalies: this makes it less expensive in terms of calculations.

Each tree is constructed randomly (node draw, threshold, random allocation) until one observation per sheet is obtained. This approach is described as ’isolation’ partitioning.

We then give a score to each observation, each tree, according to a predetermined score function. In the end, we obtain a score for each observation after averaging the score obtained. The score function established by its authors takes into account the depth of the node: the lower the depth, the higher the score and therefore the more the observation is considered as an anomaly.

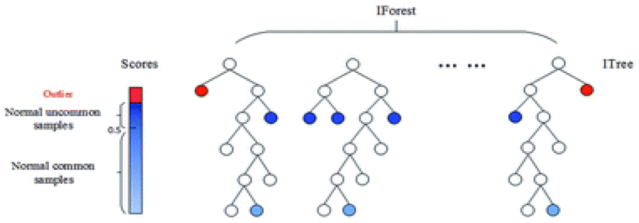


Fig. 4. Possible clustering with Isolation Forest algorithm.

The algorithm requires two parameters:

- ***”Sub-sampling”***: The number of observations to be taken from the data set to form each basic estimator.

- ***”Number of trees”***: The number of trees, or basic estimators of the set.

The author’s studies [LTZ08] have shown that Isolation Forest detection accuracy converges quickly with a very small number of trees (100 would be sufficient); and it would take a small subsampling to get a high detection accuracy with high efficiency. According to him, empirically, setting this parameter to 2^8 or 256 usually provides enough informations to perform the detection of anomalies in a wide range of data. His study showed that the detection performance of the prediction model would be near optimal at these parameter values, and would be quasi-invariant for larger parameter values.

D. *Restricted Boltzmann Machines*

Introduced by [EHAJS85] in 1985, a Boltzmann machine (also called stochastic Hopfield network with hidden units) is a type of stochastic recurrent neural network (included in the Markov random field).

They are named after the Boltzmann distribution in statistical mechanics, which is used in their sampling function. That’s why they are called ”energy based models” (EBM).

They were one of the first neural networks capable of learning internal representations, able to represent and solve difficult combinatoric problems.

Basically, a Boltzmann machine consists of two layers : a layer of neurons that receives the input, and a layer of

hidden neurons. Assuming that the neurons of the same layer are independent of each other, this configuration is called a Restricted Boltzmann Machine (RBM).

Energy-based probabilistic models define a probability distribution through an energy function, as follows:

$p(x) = \frac{e^{-E(x)}}{Z}$.

The normalizing factor Z is called the partition function by analogy with physical systems.

$Z = \sum_x e^{-E(x)}$

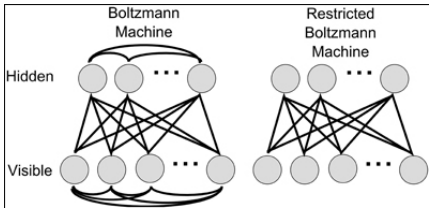


Fig. 5. Boltzmann Machines and Restricted Boltzmann Machines model.

In machine learning, the Restricted Boltzmann Machine, originally invented by Paul Smolenski in "Harmony Theory", 1986, is a type of artificial neural network for unsupervised learning. An RBM is a Generative stochastic artificial neural network that can learn a probability distribution from a training data set (input).

Theory

Basically, RBM is a Markov Random Field model associated with a two-layer undirected graph :

- The Visible Layer (the first or input layer)
- The Hidden Layer (the second layer)

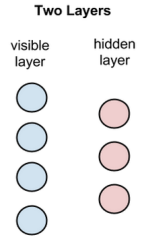


Fig. 6. RBM model.

There is no output layer for the model since the transformed input at the hidden layer is feeded as input.

It consists of n visible units $v = (v_1, ..., v_n)$ to represent the observed data and m hidden units $h = (h_1, ..., h_m)$ to capture patterns, relation between the observed variables. Both units are binary variables : it takes generally values 0 or 1.

The two layers are fully connected through a set of weights W and biases b, c and there is no connection between units of the same layer. The learning of RBMs is modelled as follows:

Phase 1 : Forward Pass or "Construction"

Each hidden node receives the four inputs multiplied by their respective weights. When signals propagate from visible to hidden, the input layer (i.e. the data sample) will be multiplied by the matrix W, or weights. The sum of those products is again added to a bias b, and the result is passed through the activation algorithm (sigmoid function to be within 0 and 1), producing one output for each hidden node.

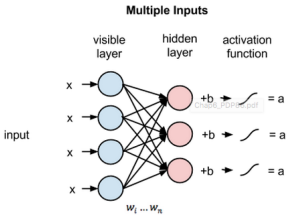


Fig. 7. Forward pass or Construction phase $f((W * x) + b) = a$

Phase 2 : Backward Pass or "Reconstruction"

In the reconstruction phase, the hidden layer activation become the input in a backward pass. They are multiplied by the same weights W, and the sum of those products is added with visible biases c, will go through the activation function and the output of those operations is a reconstruction; i.e. an approximation of the original input.

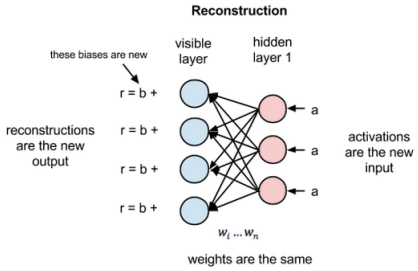


Fig. 8. Backward pass or Reconstruction phase $f((W * a) + c) = r$

We know that an RBM is a Generative stochastic artificial neural network that can learn a probability distribution from a training data set (input). Reconstruction is making guesses about the probability distribution of the original input.

The reconstruction error is the difference between the values of r and the input values x, and that error is then backpropagated against the RBM's weights, iteratively during the learning process until an error minimum is reached.

Intuitively, we can understand that , during training, the model is adjusting its weights W such that it learns to approximate the original data. Indeed, it could best approximate the training data (original data) distribution p with its reconstruction distribution q :

RBM's optimization algorithm attempts to minimize the reconstruction error and produce a close approximation of the original input. More details about leaning RBM can be found in papers [FI12].

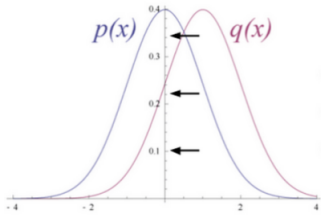


Fig. 9. RBM Learning process.

The RBM is based on an energy-based probabilistic model.

The joint probability distribution of the configuration (v,h), v visible units and h hidden hunits is defined as follows :

$$p(v,h;\theta) = \frac{e^{-E(v,h;\theta)}}{Z}.$$

The normalizing factor Z is called the partition function by analogy with physical systems.

$$Z = \sum_{v,h} e^{-E(v,h;\theta)}$$

Gaussian-Gaussian RBM (GGRBM) extends the RBM by replacing the initial binary visible and hidden units with real-valued ones that have Gaussian noise to model real-valued data. The energy function E of GGRBM is defined as follows :

$$E(v,h;\theta) = \sum_{i \in vis} \frac{(v_i - a_i)^2}{2\sigma_i^2} + \sum_{j \in hid} \frac{(h_j - b_j)^2}{2\sigma_j^2} - \sum_{i,j} \frac{v_i h_j w_{ij}}{\sigma_i \sigma_j}$$

with a and b the biases, here we set the variance the variance to 1.

The free energy [Fis14] of a visible vector v is the energy that a single configuration would need to have in order to have the same probability as all of the configurations that contain v :

$$e^{-F(v)} = \int_h e^{-E(v,h;\theta)}$$

i.e.

$$F(v) = \frac{||v-a||^2}{2} - \frac{n\log(2\pi)}{2} - \frac{||v^tW||^2}{2} - v^tWb$$

with n the number of hidden units (variance σ set to 1).

Application in fraud detection

IV. Experimental study

We seek to compare our semi-supervised deep-learning model (RBM) with the unsupervised clustering methods (DBSCAN, Isolation Forest and OCSVM).

Our study is based on the following procedure, for each method :

- Use a labelled data set.
- Remove the labels.
- Apply the algorithm in question.
- Evaluate the performance of the partitioning, comparing it to the labelled data.

Our study will be carried out on Python, using the following packages : scikit-learn (clustering algorithms) and pytorch (restricted boltzmann algorithm).

A. *Materials and Methods*

Dataset

Our experiment is based on the 'Credit Card Fraud Detection' dataset imported from Kaggle.

This dataset shows the transaction that occurred in two days with 284.807 transactions and 492 frauds (labelled dataset), ie. 0.172% of all are fraud and the rest are genuine : the dataset is highly imbalanced. The dataset is composed of 30 features, 28 of them have been-PCA transformed due to confidentiality reason.

For machine learning algorithms to be able to process categorical variables, which can be in numerical or text form, they must first be transformed into a numerical representation.

A subset of 80,000 transactions is extracted from the whole set, and the RBM model is trained with it. For the rest, we consider two subsets, so we can compare the robustness of the methods :

- The 'Small' Dataset : 15.492 transactions (including the 492 anomalies).
- The 'Big' Dataset : 204.807 transactions (including the 492 anomalies).

Parameters Choice

First of all, it is essential to analyze our dataset before applying clustering methods. Especially for methods based on distance or density measurement (DBSCAN in particular).

*1) **Data Analysis**:* We first experiment a data analysis on the 'small' data set. We start by studying the matrix of observation distances (Euclidean distance), including the minimum distance (or distance to the nearest neighbour) and average distance of each point :

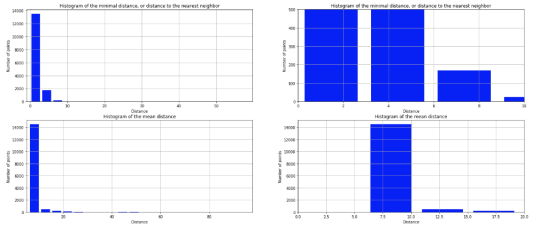


Fig. 10. Distance matrix of the dataset of 15492 observations.

We note that the majority of observations are located at a distance of at most 6 units from their nearest neighbour. Thus, we can assume that observations with a minimum distance of more than 6 are either in a low-density region or remote from the others. Therefore 6 would be a reasonable distance for the DBSCAN epsilon parameter.

In addition, we can see that the majority of observations are located at an average distance of less than 10 units from other observations. Thus, we can hypothesize that observations that have an average distance of more than 10 units are either in ”remote” from the others or in a region of low density. We therefore arbitrarily set 1021 (number of observations with an average distance greater than 10) the number of frauds expected. Of course, this approach has been carried out in a rather broad and not very strict way, in order to be able to identify as many frauds as possible.

We will then look at the number of neighbours of each observation, in a ball of radius r=6 (the minimum distance we have chosen) :

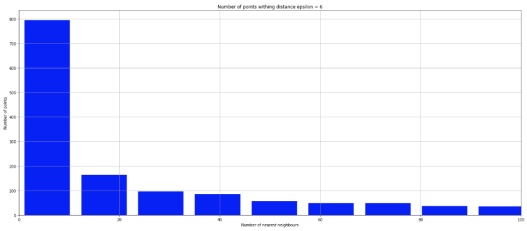


Fig. 11. Histogram of the number of neighbours in a ball with a radius of 6 units.

We can see in the graph above that nearly 1100 (less than 10% of the observations, here the number of outliers expected) have very few neighbours, between 0 and 25 neighbours in a ball of radius 6 units.

This would mean that either the points are isolated or they are located in a very low density region. This could therefore mean that the majority of outliers are among this proportion of observations. And so observations with more than 25 neighbours in the ball of radius r=6 units could be non fraudulent.

So, based on the information extracted from these histograms, we will set the DBSCAN parameters such as **epsilon = 6, minPts = 25** and check our approach. **The expected number of outliers remains at 1021 outliers**.

Nota Bene :

The histogram below shows us that the majority of outliers are part of the portion of observations considered to be anomalies in our previous approach (less than 10 neighbours):

B. Choosing parameters, improvement and implementation

Following this data analysis, we can set the parameters as follows: **epsilon = 6, minPts = 25** and **the expected number of outliers remains at 1021 outliers**. The clustering algorithms used are on the *sklearn* package (scikit-learn.org). We then want to optimize and/or improve the algorithms studied.

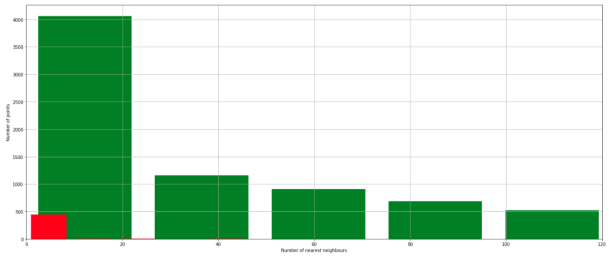


Fig. 12. The red bars represent the’outliers’, while the green bars represent the ’inliers’.

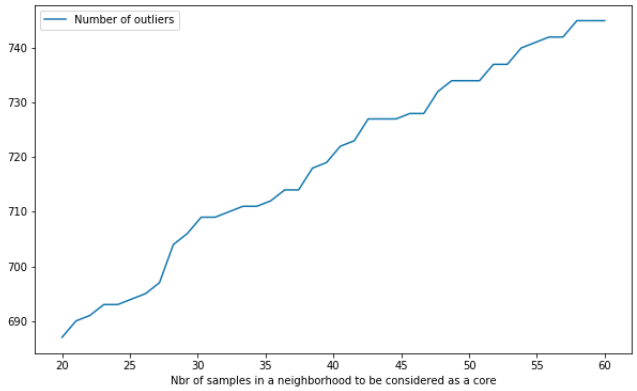
1) Parameterization and optimization of the DBSCAN algorithm: We set the following parameters in our previous study:

- **Epsilon** = 6 units. The maximum distance between two neighbors so that they are considered as in the same cluster. If the distance to its nearest neighbour is greater than epsilon, i.e. 6 units, then it is considered outlier.

- **MinPts** = 25 neighbors. Either each point must have at least 25 points in its Epsilon neighborhood (of 6 units) to form a cluster.

We consider the most important cluster to be fraudulent, the other clusters are grouped together and judged outliers.

Verification of the sensitivity of the algorithm according to the MinPts parameters between 10 and 60 (Epsilon fixed at 6 Units):



Number of outliers detected according to the value of the MinPts parameter.

We can therefore see that from MinPts=25, the number of outliers detected is about 700, close to the expected number. The number of outliers then increases when the parameter MinPts increases, which reassures us in our choice of parameters.

2) Setting up and optimizing Isolation Forest: We know that the algorithm is based on two parameters: the’sub-sampling’ and the’number of trees’.

We have chosen to keep all the variables available (so no sub-sampling) and set the number of trees to 250.

In our study, we use the score function implemented on the *scikit-learn* algorithm, according to which observations with a negative score (and therefore less than 0) are considered outliers, thus classified in the 'fraud' category.

The algorithm available on scikit-learn also requires another important parameter: *contamination*,which is the proportion of outliers in the dataset we're trying to obtain. It is set according to the number of outliers expected (here 1021).

3) Settings and optimization of the OCSVM: We saw that the most significant parameters are ν and γ.

We set ν equal to the number of outliers expected (here **1021**), and vary γ :

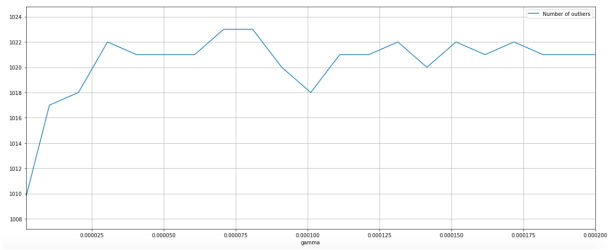


Fig. 13. Number of outliers detected according to the value of the parameter γ.

We're looking for a γ as small as possible, but one that meets the expected number of outliers. So we set γ = **0.000025**, because we can see that this value is quite small and that from this value the number of outliers is set to the value of ν, 1021 outliers expected.

4) Settings and optimization of the RBM: We fix 15 the number hidden units, learning rate of 10^{-4} and a momentum of 0.95. We first normalize each component of the data set to have zero mean and unit variance. We use gaussian visible and hidden units, more informations to tuning RBM is fully covered in Geoffrey Hinton's notes [Hin10].

Evaluation Metrics

In this study, we use three effective performance indices of classification :

- **The Recall** : the rate of true positives, i.e. the proportion of positives that have been correctly identified. Here: This is the ability of our model to detect all frauds.
  The 'recall' formula is as follows :
  $$Recall = \frac{TP}{TP + FN}$$
  .
- **Precision**, i.e. the proportion of correct predictions among the points that have been predicted positive. Here: the ability of our model to trigger an alarm only for a real fraud.

The 'precision' formula is as follows :

$$Precision = \frac{TP}{TP + FP}$$

.

C. Experimental Result and Discussion

Comparison between the proposed model (RBM) and three studied models (DBSCAN, OCSVM and Isolation Forest) :

Dataset	Evaluation Metrics	DBSCAN	OCSVM	Isolation Forest	RBM
'Small' Dataset	Precision	**0.55**	0.29	0.41	0.36
	Recall	**0.84**	0.59	0.83	0.83
'Big' Dataset	Precision	-	0.10	0.12	**0.15**
	Recall	-	0.60	0.71	**0.84**

The data analysis implemented seems to be effective for the DBSCAN, OCSVM, and Isolation Forest methods, especially for small data sets. It would be interesting to deepen this study and establish a stricter and more precise choice of parameters in order to improve precision.

DBSCAN seems to perform very well on small data set, however the complexity of the algorithm is really important (complexity $O(n^2)$ at least), which is a huge inconvenient for larger data set (couldn't be able to run the algorithm : the CPU crashes starting from the 30.000 iteration of the algorithm (memory usage too much important).

RBM seems to perform very well on large and small data set (detects almost 84% of the frauds, ie. 413 over 492 frauds), which is not the case for clustering algorithms used : very good with small data sets but not that good with larger ones (poor precision mostly) : lack of robustness.

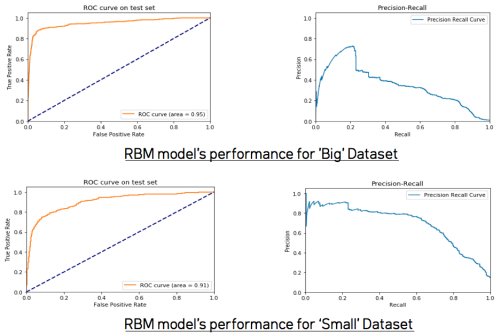


Fig. 14. ROC curve and precision-recall for the RBM.

V. Conclusion and Future Work

RBM is a semi-supervised technique which is a very good tool for real-time fraud detection (possibility to train the model with new genuine data and improve the accuracy of the model), while the statistical learning algorithms used are built for an unsupervised outlier detection.

Future work :

- Tune the hyper parameters of the RBM : make it more robust.

- It would be interesting to conduct future work on other datasets : verify the generalization of the results presented by this study.

- It would also be interesting to combine our methods (using ensemble learning) : keeping the advantages of each algorithm to build a better model (for example Isolation Forest and RBM for fully unsupervised anomaly detection).

REFERENCES

[EHAJS85] Geoffrey E. Hinton, H. Ackley, and Terrence J. Sejnowski. A learning algorithm for boltzmann machines. *Cognitive Science*, 9(1):147–169, 1985.

[FI12] Asja Fischer and Christian Igel. "an introduction to restricted boltzmann machines." progress in pattern recognition, image analysis, computer vision, and applications. springer berlin heidelberg. page 14, 2012.

[Fis14] Asja Fischer. Training restricted boltzmann machines. *Faculty of Science, University of Copenhagen*, 2014.

[Hin10] Geoffrey Hinton. A practical guide to training restricted boltzmann machines. 2010.

[LTZ08] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. 2008.

[Res18] R.S. Reshma. Deep learning enabled fraud detection in credit card transactions. 2018.

[UFAA13] Fiore Ugo, Palmieri Francesco, Castiglione Aniello, and De Santis Alfredo. Neurocomputing. 2013.