Building Machine Learning Models for Fraud Detection in Customs Declarations in Senegal

DJAMAL ABDOUL NASSER SECK
Faculty of Science and Technology,
Cheikh Anta Diop University of Dakar,
BP 5005 Dakar-Fann,
SENEGAL

Abstract: - To improve the customs declaration control system in Senegal, we propose fraud risk prediction models built with machine learning methods such as Neural Networks (MLP), Support Vector Machine (SVM), Random Forest (RF) and eXtreme Gradient Boosting (XGBoost). These models were built from historical customs declaration data and then tested on a part of the data reserved for this purpose to evaluate their prediction performance according to the metrics of accuracy, precision, recall, and F1-Score. The RF model proved to be the more performant model and is followed, in order, by the XGBoost model, and the MLP and SVM models.

Key-Words: - fraud detection, customs declarations, machine learning, supervised method, model, prediction, binary classification.

Received: June 14, 2023. Revised: January 25, 2024. Accepted: February 13, 2024. Published: March 26, 2024.

1 Introduction

Fraud detection in customs declarations is of great importance for a country. Indeed, customs fraud creates economic and security risks for a country because it can lead to loss of financial revenue or compromise national security through the entry of illicit or dangerous goods. In Senegal, the Customs Administration, which is in charge of collecting revenue and combating fraud, has set up an automated system to manage the risks of fraud in declarations during the importation of goods. This system is essentially based on two techniques: customs intelligence and targeting of risky declarations. Intelligence is the process of obtaining information about fraud activity through certain people called informants. This information is then used to obtain customs intelligence. However, the use of these techniques, which are essentially based on the human perception of fraud, involves a lot of subjectivity in the system and can distort the targeting of risky transactions. Thus, to bring more objectivity and accuracy to the current fraud risk management system, we propose in this paper a fraud detection solution based on artificial intelligence with the use of machine learning models.

In the rest of this paper, we will first talk about the context and the problem of our proposal. Next, we're going to talk about machine learning, covering some general information and presenting some of its methods. Then, we will present the models for detecting fraud in customs declarations that we propose by explaining the methodology of their construction and presenting the results of their performance tests. We will finish with a conclusion and perspectives.

2 Context and Issues

The fraud risk management system set up by Senegalese customs is essentially based on the intelligence and knowledge of the customs officer in terms of fraud. It is a decision support tool that directs verification officers to the appropriate types of controls. Thus, the collected information, combined with the experience of the customs officer, makes it possible to identify risk criteria. Subsequently, these criteria are prioritized according to their impact on fraud, and risk profiles are defined by

combining the values of the identified risk criteria. The defined profiles will serve as the basis for targeting risky declarations. Thus, the declarations concerned by this targeting will systematically be subject to a physical check, while the declarations deemed to be safe will only be subject to a documentary check. It should be noted that the results of the checks carried out by the auditors are entered into the information system. As a result, a fraud database is created. However, this database is not sufficiently exploited to create, for example, a model for calculating the risk of fraud on customs declarations. Thus, the current risk management system of the Senegalese Customs presents a problem of objectivity in the assessment of the risk of fraud insofar as the targeting of risk declarations is essentially based on the information collected by the verification officers as well as their knowledge of the history of fraud.

In this context, to overcome the inadequacies of the Senegalese customs' fraud risk management system, we propose to exploit the collected data to build fraud detection models using supervised machine learning methods.

3 Machine Learning Methods

In this section, we give a definition of machine learning and present some of its methods that we used to build the models for detecting fraud in customs declarations.

3.1 Definition of Machine Learning

Machine Learning, [1], is a sub-field of artificial intelligence whose goal is to give machines the ability to discover patterns in data for decision-making using learning methods that can be supervised in the case of prediction problems, [2], [3], or unsupervised in the case of other types of problems.

A supervised machine learning method learns from the data the mappings between inputs represented by variables X1, X2, .., XP and outputs represented by a variable Y to find a function φ such that $Y = \varphi(X1, X2, .., XP)$ that will serve as a model to predict Y for any new input data ω given its attributes $X1(\omega)$, $X2(\omega)$, .., $XP(\omega)$.

The variable Y to be predicted is called the dependent variable and the variables X1, X2, ..., XP are called independent variables.

The prediction task is a classification if Y is a qualitative variable and a regression if Y is a quantitative variable.

In this paper, we propose binary classification models to predict whether or not a customs declaration is fraudulent. These models are built from historical customs declaration data with the following supervised machine learning methods: Multilayer Perceptron, Support Vector Machine, Random Forest, and Extreme Gradient Boosting.

3.2 Multilayer Perceptron

The multilayer Perceptron, [4], is a type of artificial neural network used for classification or regression tasks. An artificial neural network is a machine learning model that is inspired by the functioning of the human brain and is composed of artificial neurons organized in layers and connected by weighted connections. Each artificial neuron is a computational unit that receives input data through its input connections, adds a value called activation threshold or bias, and applies an activation function to give an output.

In a multilayer perceptron, there is an input layer, one or more hidden layers, and an output layer. Every neuron of a layer is connected to all neurons in the next layer and there are no connections between neurons in the same layer. The connections between neurons are characterized by weights that are real numbers.

The multilayer perceptron is a feed-forward neural network, which means that information flows from the input layer to the output layer. Neurons in the input layer have no bias or activation function. They only receive the input data and send them to the neurons in the first hidden layer that processes them with their activation function, and then, in turn, send their outputs to the neurons in the next layer, and so on until the output layer. The outputs of the neurons in this last layer are the result of the prediction with the neural network and depend on the weights of the connections between the neurons in the network.

Before using the multilayer perceptron to predict the output for input data, it must first be trained with a set of input-output data called a training set to find the optimal weights that enable to have good prediction results. This training phase carried out with the back-propagation algorithm, [4], [5] and the gradient descent optimization method, consists of iteratively modifying the weights of the connections between neurons to minimize the prediction error.

3.3 Support Vector Machines

Support Vector Machines, [6], [7] is a machine learning method whose operating principle consists of reducing the problem of classification to that of finding a hyperplane that will separate, in the space of characteristics, the examples belonging to different classes and maximize the distance between these classes. Such a hyperplane is called the optimal hyperplane and the distance between the classes is called the margin. The closest examples, which alone are used for the determination of the optimal hyperplane, are called support vectors.

Among the SVM models, we can distinguish between linear SVMs and nonlinear SVMs. Linear SVMs are the simplest because they make it easy to find the optimal hyperplane. For nonlinear SVMs, the idea is to achieve, via a kernel function, [8], a nonlinear transformation of the data space to allow a linear separation of the examples in the new space.

3.4 Random Forest

Random Forest, [9], is a special case of bagging (bootstrap aggregating), [10], which is an ensemble method whose principle is to combine forecasts from several independent models to reduce the variance and therefore the error of prediction. These models are built from bootstrap samples obtained by random draw with replacement from the same data set. In the case of the Random Forest method, each of these models is a decision tree, [1], [11], [12], constructed by the recursive partitioning of a bootstrap sample. Each partitioning is based on a test on a cut-off variable chosen from a subset of input variables selected at random. The final prediction for a given example is the majority of the predictions of the different trees in the case of classification and the average in the case of regression.

3.5 Extreme Gradient Boosting

Extreme Gradient Boosting, [13], is a special case of boosting, [14], which is an ensemble method whose principle is to sequentially aggregate weak prediction models (weak learners) into a performing model. These models are built successively on different weight distributions of the examples of the training sample, each model being trained to correct the errors of those preceding it. Extreme Gradient Boosting builds decision tree models and is an optimization of the gradient boosting method which is a high-performance boosting method that uses the gradient of the loss function for the calculation of example weights when building each new model.

4 Building of the Detection Models of Customs Fraud

In this section, we first present the data we use for the building of our fraud detection models in customs declarations. We also describe the pre-processing operations we apply to these data to improve the quality of the models. Then, we explain the building of these models on a part of the data used as the training set. Then, using performance metrics, we compare the performance of these models on the other part of the data used as a test set.

4.1 Presentation of the Data

The dataset we use to build our fraud detection models consists of 25254 examples of customs declarations characterized by the following 11 variables:

- REGIME : the Customs procedure,
- PRODUIT : the imported product,
- MODE_DE_CONDITIONNEMENT : the packaging mode of the product ,
- CODE_ORIGINE : the origin of the goods,

- CODE_PROVENANCE : the provenance of the goods,
- VALEUR_CAF : the value of the goods,
- POIDS_NET : the weight,
- CODE_DESTINATAIRE : the importer of the goods,
- CODE_DECLARANT : the customs declarant,
- MODE_DE_PAIEMENT : the method of payment,
- FRAUDE : the result of the inspection.

FRAUDE variable, which represents the result of the inspection, is the dependent variable. It is a class variable with two distinct values: 0 which means no fraud and 1 which means fraud. Of the other variables, which are the independent variables, only VALEUR_CAF and POIDS_NET are quantitative variables. The others are qualitative variables. Figure 1 shows the dataset.

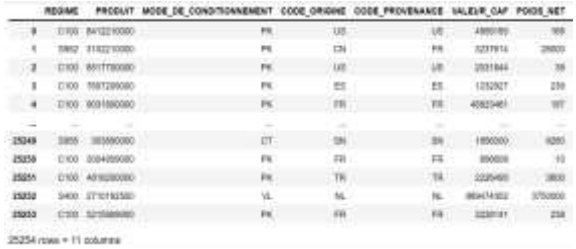

Fig. 1: The dataset

The info() method of the pandas package gives information about the dataset such as the number of examples, the list of variables with their types, and their respective non-zero value counts.

For our dataset, there are 25254 examples and as many non-zero values for each of the variables, which means that there are no missing values as shown in Figure 2.

```
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 25254 entries, 0 to 25253
Data columns (total 11 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   REGIME                  25254 non-null  object
 1   PRODUIT                 25254 non-null  object
 2   MODE_DE_CONDITIONNEMENT 25254 non-null  object
 3   CODE_ORIGINE            25254 non-null  object
 4   CODE_PROVENANCE         25254 non-null  object
 5   VALEUR_CAF              25254 non-null  int64
 6   POIDS_NET               25254 non-null  int64
 7   CODE_DESTINATAIRE       25254 non-null  object
 8   CODE_DECLARANT          25254 non-null  object
 9   MODE_DE_PAIEMENT        25254 non-null  object
 10  FRAUDE                  25254 non-null  int64
dtypes: int64(3), object(8)
memory usage: 2.1+ MB
```

Fig. 2: Information on the data

4.2 Data Sampling

To build our fraud detection models, we divide the data into a training set and a test set. The training set is used to build the models with the chosen machine learning methods while the test set is used to evaluate the prediction performance of the models and thus evaluate their ability to generalize to new data.

To do the data splitting, we use the train_test_split( ) method of the model_selection module included in the Scikit-learn, [15], python package to select 67% of the data to obtain a training set of 16920 examples and 33% to obtain a test set of 8334 examples.

4.3 Balancing the Class Distribution in the Training Set

Analyzing the distribution of classes (FRAUDE = 0 and FRAUDE = 1) in the training set, we observe a significant imbalance, with about 98.55% of the examples in class 0 and only 1.45% in class 1 as shown in Figure 3.

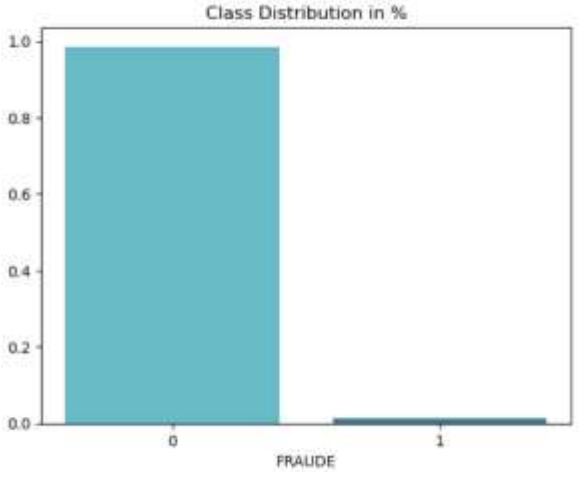

Fig. 3: Class distribution in the training set

It is important to solve this class imbalance problem before building our models from the training sample because it can lead to a bias in the models which will then tend to predict the majority class (FRAUDE = 0).

The class imbalance can be corrected with the oversampling technique, which consists of artificially increasing the number of examples of the minority class. For this, we use the SMOTENC( ) method of the imblearn.over_sampling module , which gives us a learning set of 30015 examples, of which 55.56% are from class 0 and 44.44% are from class 1 as shown in Figure 4.

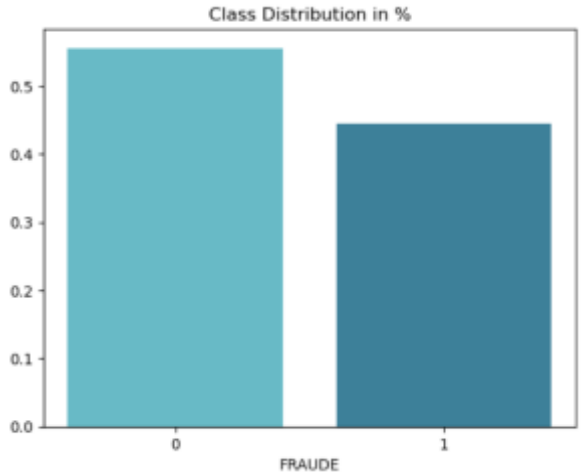

Fig. 4: Class distribution in the training set after over-sampling

4.4 Numerical Encoding of the Qualitative Variables

The python implementations, [15], [16] of the machine learning methods we use to build our models mostly require the data to be numerical. Thus, it is necessary to perform a numerical encoding of the qualitative variables of our dataset.

The type of numeric encoding that is appropriate for the qualitative variables in our dataset is one hot encoding because they are nominal qualitative variables whose values are unordered categories.

Applying one hot encoding to a nominal qualitative variable that has n distinct categories consists of replacing it with n corresponding binary numeric variables. For each example in the dataset, the binary variable corresponding to the observed category is set to 1 and the other binary variables are set to 0. For example, if a nominal qualitative variable Xj has 3 distinct categories a, b, and c, then it will be replaced by 3 binary variables Xj_a, Xj_b, Xj_c as shown in Figure 5.

	X1	X2	. . .	Xj	. . .	XP
(1)				a		
(2)				b		
(3)				b		
(4)				a		
(5)				c		

⇩

	X1	X2	. . .	Xj_a	Xj_b	Xj_c	. . .	XP
(1)				1	0	0		
(2)				0	1	0		
(3)				0	1	0		
(4)				1	0	0		
(5)				0	0	1		

Fig. 5: One hot encoding of a qualitative variable

But if we apply this encoding technique directly to the qualitative variables in our dataset, the number of binary variables generated will be very high, because these variables mostly have high numbers of distinct categories.

To avoid this problem of exploding the number of variables, we limit the one hot encoding by replacing each of the qualitative variables with a maximum of ten binary variables corresponding to its ten most frequent categories. The other categories will be grouped under a new category that will be dropped. So, for any example in the dataset, if the observed category is one of the most frequent categories, then the corresponding binary variable will be set to 1 and the others will be set to 0. If the observed category is one of the other categories, then all binary variables will be set to 0.

4.5 Scaling of the Quantitative Variables

After the numerical encoding of the qualitative variables, we need to scale all of our variables to avoid having biases in the models we want to build with our training set. To do this, we use

min-max normalization to transform the values of these variables into values between 0 and 1.

4.6 Building of the Models

After the data pre-processing, we build our fraud detection models using the machine learning methods presented above, namely: Neural Networks (MLP), Support Vector Machine (SVM), Random Forest (RF), and eXtreme Gradient Boosting (XGBoost).

We use the Scikit-learn, [15], python package and XGBoost, [16], library to sample the data and build and test the models.

Sampling consists of dividing the data into two samples: a training sample for building the models and a test sample for testing the models.

For the construction of the models, the FRAUDE characteristic is the dependent variable. It is a qualitative variable whose values are the classes to be predicted: 0 and 1. The other characteristics are the independent variables.

Each built model is tested to evaluate its prediction performance.

4.7 Prediction Performance of the Models

To evaluate the prediction performance of our models, we use the metrics of accuracy, precision, recall, and F1-score.

The accuracy of a model is the proportion of correct predictions over all the predictions of the classes for the examples of the test set.

Accuracy, recall, and F1-score measure the predictive performance of a model relative to one of the classes in the test sample that is considered the positive class. Examples in this positive class are positive examples, while those in the other classes are negative examples.

The precision of a model is the proportion of positive examples predicted as positive (true positives) over all the examples predicted as positive. It measures the model's ability not to predict the positive class for a negative example.

The recall of a model is the proportion of examples predicted as positive over all the positive examples. It measures the model's ability to predict the positive class for a positive example.

A good model is one with high values of accuracy and recall. However, as the accuracy increases, the recall decreases and vice versa. To solve such a dilemma, we use the F1-score, the harmonic mean of the accuracy and the recall. An optimal value of the F1-score corresponds to both an optimal value of the precision and an optimal value of the recall.

Accuracy, recall, and F1-score are calculated relatively to each of the classes of the test set. After, we find the averages of the results weighted by the number of examples in the classes to get the overall values of these metrics.

Table 1 shows the precision, recall and F1-score values of our fraud detection models relative to the classes 0 (no fraud) and 1 (fraud) and the supports (numbers of examples) of those classes.

Table 1. Precision recall and F1-score of the models relative to the classes

	Class	Support	RF	SVM	XGBoost	MLP
Precision	**0**	8217	0.99	0.99	0.99	0.99
	1	117	0.14	0.06	0.12	0.06
Recall	**0**	8217	0.97	0.90	0.95	0.90
	1	117	0.37	0.46	0.50	0.44
F1-score	**0**	8217	0.98	0.94	0.97	0.94
	1	117	0.20	0.10	0.19	0.10

Table 2 shows the overall prediction performance of the models on the test set, represented by the accuracy and the averages of the precision, of the recall and of the F1- score.

Table 2. Global prediction performance of the models

Model	Precision	Recall	F1-score	Accuracy
RF	0.98	0.96	0.97	0.96
SVM	0.98	0.89	0.93	0.89
XGBoost	0.98	0.94	0.96	0.94
MLP	0.98	0.89	0.93	0.89

According to the results in Table 1 and Table 2, the random forest (RL) model is the most efficient, followed by the Extreme Gradient Boosting (XGB) model. Then, the SVM

(Support Vector Machines) model and the Perceptron Multilayer (MLP) model follow with the same performance.

5 Conclusion

In this paper, we proposed machine learning models for the detection of fraud in customs declarations in Senegal. These models were built using data from customs declarations and using the following supervised learning methods:: Multilayer Perceptron, Support Vector Machines, Random Forest, and Extreme Gradient Boosting. The model obtained with Random Forest was found to perform the best according to the performance measures we used, namely precision, recall, F1-score, and accuracy. Then follow, in order, the model obtained with Extreme Gradient Boosting and the models obtained with Multilayer Perceptron and Support Vector Machines.

In perspective, it would be interesting to combine these models to form an ensemble model that would be very efficient in fraud detection.

These models could be integrated into the Senegalese Customs' fraud risk management system to improve the efficiency of controls, and facilitate the work of customs officers.

References:

[1] T. Mitchell, *"Machine learning"*, McGraw Hill, 1997.

[2] D. A. N. Seck and F. B. R. Diakité, "Supervised Machine Learning Models for the Prediction of Renal Failure in Senegal," *2023 International Conference on Control, Artificial Intelligence, Robotics & Optimization (ICCAIRO)*, Crete, Greece, 2023, pp. 94-98, DOI: 10.1109/ICCAIRO58903.2023.00022.

[3] N. Paranoan, S. Y. Sabandar, A. Paranoan, E. Pali, I. Pasulu, "The Effect of Prevention Measures, Fraud Detection, and Investigative Audits on Efforts to Minimize Fraud in The Financial Statements of Companies, Makassar City Indonesia," *WSEAS Transactions on Information Science and Applications*, vol. 19, pp. 54-62, 2022, https://doi.org/10.37394/23209.2022.19.6.

[4] P. J. Werbos, *"Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences"*, Doctoral Dissertation, Harvard University, Cambridge, 1974.

[5] D. E. Rumelhart, G. E. Hinton, R. J. Williams (1986) , *"Learning representations by back-propagating errors"*, Nature, Vol 323, 533-536.

[6] B. E. Boser, I. M. Guyon, and V. N. Vapnik, *"A training algorithm for optimal margin classifiers"*, In Proceedings of the fifth annual workshop on Computational learning theory, pages 144–152, 1992.

[7] C. Cortes and V. Vapnik, "Support-vector networks", *Machine learning*, 20(3):273–297, 1995.

[8] N. Cristianini and J. Shawe-Taylor, "An introduction to support vector machines and other kernel-based learning methods", *Cambridge University Press*, 2000, DOI: 10.1017/CBO9780511801389.

[9] L. Breiman, "Random forests", *Machine learning*, 45(1):5–32, 2001.

[10] L. Breiman, "Bagging predictors", *Machine Learning* 24(2), 123-140, 1996.

[11] J. Quinlan, *"C4.5: Programs for Machine Learning"*, Morgan Kaufman, San Mateo, California, 1993.

[12] L. Breiman, J. Friedman, R. Olshen, C. Stone, *"Classification and Regression Trees", Wadsworth, Belmont*, California, 1984.

[13] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system", *In Proceedings of the 22nd Acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[14] R. Schapire, "The strength of weak learnability", *Machine Learning*, 5(2):197–227, 1990.

[15] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau., "Scikit-learn: Machine Learning in Python", *JMLR 12*, pp. 2825-2830, 2011.

[16] T. Chen, C. Guestrin, "XGBoost: A Scalable Tree Boosting System", *In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York, NY, USA: ACM; 2016, p. 785–94, (KDD '16), Available from: http://doi.acm.org/10.1145/2939672.2939785.

Contribution of Individual Authors to the Creation of a Scientific Article (Ghostwriting Policy)

The authors equally contributed in the present research, at all stages from the formulation of the problem to the final findings and solution.

Sources of Funding for Research Presented in a Scientific Article or Scientific Article Itself

No funding was received for conducting this study.

Conflict of Interest

The authors have no conflicts of interest to declare.