Journal of Computer and Communications, 2023, 11, 142-157
https://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

Machine Learning-Based Approach for Identification of SIM Box Bypass Fraud in a Telecom Network Based on CDR Analysis: Case of a Fixed and Mobile Operator in Cameroon

Eric Michel Deussom Djomadji[1,2], Kabiena Ivan Basile[2,3], Tchapga Tchito Christian[1], Ferry Vaneck Kouam Djoko[2], Michael Ekonde Sone[1]

[1]Department of Electrical and Electronic Engineering, College of Technology, University of Buea, Buea, Cameroon
[2]Division of Information and Communications Technology, National School of Posts, Telecommunications and Information and Communication Technologies, University of Yaoundé I, Yaoundé, Cameroon
[3]Department of Computer Engineering and Telecommunications, National Advanced School of Engineering, University of Douala, Douala, Cameroon
Email: eric.deussom@gmail.com

How to cite this paper: Deussom Djomadji, E.M., Basile, K.I., Christian, T.T., Djoko, F.V.K. and Michael, S.E. (2023) Machine Learning-Based Approach for Identification of SIM Box Bypass Fraud in a Telecom Network Based on CDR Analysis: Case of a Fixed and Mobile Operator in Cameroon. *Journal of Computer and Communications*, **11**, 142-157.
https://doi.org/10.4236/jcc.2023.112010

Received: January 29, 2023
Accepted: February 25, 2023
Published: February 28, 2023

Abstract

In the telecommunications sector, companies suffer serious damages due to fraud, especially in Africa. One of the main types of fraud is SIM box bypass fraud, which includes using SIM cards to divert incoming international calls from mobile operators creating massive losses of revenue. In order to provide a solution to these shortcomings that apply almost to all network operators, we developed intelligent algorithms that exploit huge amounts of data from mobile operators and that detect fraud by analyzing CDRs from voice calls. In this paper we used three classification techniques: Random Forest, Support Vector Machine (SVM) and XGBoost to detect this type of fraud; we compared the performance of these different algorithms to evaluate the model by using data collected from an operator's network in Cameroon. The algorithm that produced a better performance was the Random Forest with 92% accuracy, so we effectuated the detection of existing fraudulent numbers on the telecommunications operator's network.

Keywords

CDR, Fraud Detection, Machine Learning, Voice Calls

1. Introduction

Cameroon's economy pays a high price for international telephone calls made

through the SIM Box fraud system. In 2015, the loss of revenue reached 22.2 billion FCFA. That is 18 billion for the 4 local telephone operators, namely CAMTEL, MTN Cameroon, Orange Cameroon, and Nexttel. This is the bill for 100 million minutes of calls made from abroad. As for the state, it loses 4.2 billion in terms of uncollected taxes. In 2014, the overall losses were CFAF 9.3 billion. Operators bore 7.5 billion and the state 1.8 billion [1]. The SIM Box consists of making an international call for a local call via the internet. The receiver sees a local number displayed while the call comes from outside. Commonly especially in Africa and Asia, this fraud causes financial losses of between 2.3 and 7 billion dollars worldwide [2]. Fraud is a major problem for mobile network operators worldwide, costing them more than 38 billion U.S. dollars per year [3]. In many countries, the rate for routing international calls (ITR) is considerably higher than the rate for routing local calls. Fraudsters make considerable profit by bypassing the routing of the licensed international operator to terminate calls in the country. As a result, fraudsters pay the local rate, which is lower than the International Routing Rate (ITR). This practice is illegal in most countries and is an important issue for many operators because of the associated loss of revenue.

In the context of this research, we worked on the case of a fix and mobile operator in Cameroon. The operator has implemented several solutions to reduce SIM box frauds, but so far these methods of fighting SIM Box fraud are not effective and the operator continues to suffer financial losses. The existing solution used by the operator does not allow for obtaining a real time analysis of CDRs for the detection of fraud by SIM Box. However, these security measures have many limitations in terms of real time analysis of CDRs and detection of fraud. Therefore, we propose a Machine Learning based approach for real-time CDR analysis and efficient SIM Box fraud detection. The proposed method can be used in every telecommunications network, we apply it on this network operator as a case study given that the real data was collected there.

In SIM Boxes, local SIM cards are used for rerouting/bypassing international calls from mobile network operators then transfer them over the Internet and deliver them back by means of VoIP gateway device called SIM-Box, as local calls to the operator's cellular network [4]. **Figure 1** and **Figure 2** respectively present the case of a normal international call and the case of a fraud using a SIM box.

A number of researches have been conducted using different tools and techniques or methods to solve the problem related to SIM-Box detection using machine-learning techniques.

Figure 1. Legitimate route of international call, adopt from [4] [5].

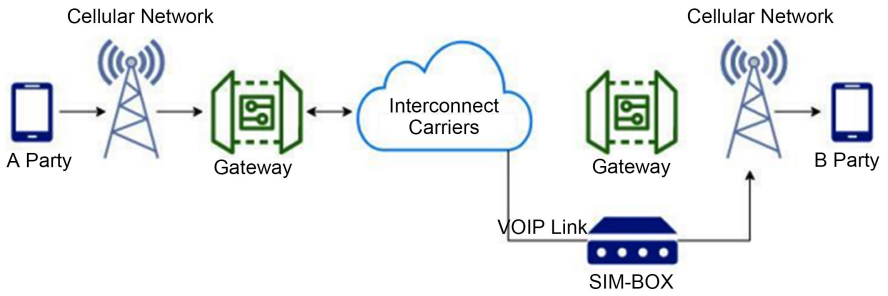


Figure 2. SIM-Box fraud rout of international call, adopt from [4] [6].

D. I. Ighneiwa and H. S. Mohamed in [7] used unsupervised learning algorithms to cluster SIMs to get insights on how they could improve the designed algorithm; different models were trained to detect SIMs used in SIM boxes.

A. Krenker, M. Volk, U. Sedlar, J. Bešter, and A. Kos in [8] prove that using a bidirectional neural network (bi-ANN) to predict generic cell phone fraud in real time yielded a high percentage of accuracy. The bi-directional neural network is used to predict the time series of subscriber call duration to identify any unusual behavior. The results show that the Bi-ANN is able to predict these time series with a rate of 90% in an optimal network configuration.

A. H. Elmi, R. Sallehuddin, S. Ibrahim, and A. M. Zain in [9] used a set of 234,324 calls made by 6415 subscribers of a single cell ID over a two-month period for analysis. The dataset included 2126 fraudulent subscribers and 4289 normal subscribers, equivalent to two-thirds of legitimate subscribers and one-third of fraudulent SIM boxes. The researchers extracted 9 features, such as total number of calls, total number of minutes and average number of minutes, etc. They then used the extracted features to train an artificial neural network (ANN) classifier. They found that the best architecture was the one with two hidden layers, each with five hidden neurons, with a learning rate of 0.6. Accuracy reached 98.7% with only 20 counts wrongly classified as false positives.

DEUSSOM Eric *et al.*, in [10] detect fraud by analyzing CDRs and internet traffic. The Differential Privacy model was used to encrypt users' personal information, and the k-means algorithm and DBSCAN were used here to group users into different clusters. Using a plane representation, they were able to visualize the users that are suspected of fraud. These were the users who were very far away from the different cluster centres.

S. Subudhi and S. Panigrahi in [11] presented a new approach to detect fraudulent activities in mobile telecommunications networks using possibilistic fuzzy c-means clustering. First, the optimal values of the clustering parameters were estimated experimentally. The modelling of the subscriber behaviour profile is then performed by applying the clustering algorithm on two relevant call features selected from the subscriber's historical call records. All symptoms of intrusive activity are detected by comparing the most recent call activity with their normal profile. Through the following authors presented, we can see that machine learning can be used in many use cases, like fraud detection, network maintenance [12] and so one. The rest of this paper is organized as follows: in

section 2, the materials and methods are presented followed by the results and comments in section 3 and finally a conclusion.

2. Materials and Methods

We used machine learning to analyze CDRs to develop a collaborative model capable of identifying SIM Box fraud using three machine learning algorithms: Random Forest, SVM and XGBOOST. Since the CDR data is labeled (data belonging to a fraudster and a non-fraudster respectively), the classification method is the best way to distinguish between fraudulent and non-fraudulent numbers. These three algorithms have many advantages; they are simple, fast and easy to understand and above all they give a result with good accuracy.

In this work, in order to explore data that has been shown to work well with unbalanced datasets, we implemented three learning algorithms.

2.1. The Random Forest Algorithm

The Random Forest algorithm is a classification algorithm that reduces the variance of the predictions of a single decision tree, thus improving their performance, by combining multiple decision trees in a bagging approach. In its most classical form, it performs parallel learning on multiple randomly constructed decision trees trained on different subsets of data. The random forest algorithm is known to be one of the most efficient "out-of-the-box" classifiers (*i.e.*, requiring little data pre-processing) [13]. The random forest algorithm works by 4 steps that won't be presented again here. **Figure 3** presents an illustration of the random forest.

2.2. The SVM Algorithm

Support vector machines (SVM) are a set of supervised learning techniques designed to solve problems. They were developed in the 1990's from the theoretical considerations of Vladimir Vapnik on the development of a statistical theory of learning: the Vapnik-Chervonenkis theory. They were quickly adopted for their ability to work with high-dimensional data, the low number of hyperparameters, their theoretical guarantees, and their good results in practice [15].

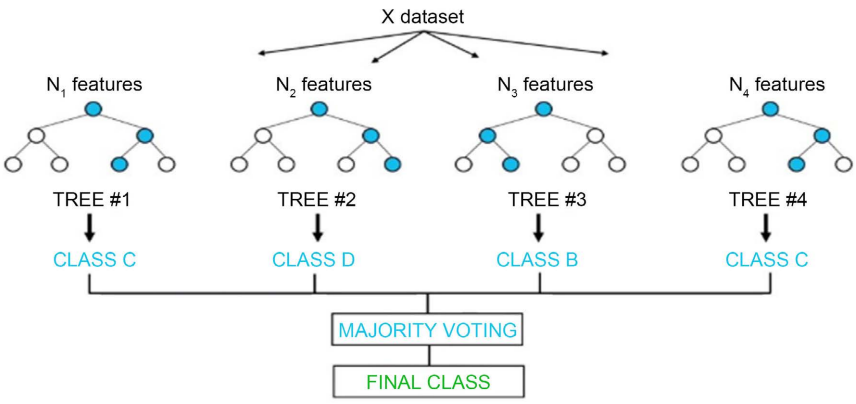


Figure 3. Random forest [14].

2.3. The XGBoost Algorithm

XGBoost was originally started as a research project by Tianqi Chen in the Distributed (Deep) Machine Learning Community (DMLC) group. XGBoost is a popular and efficient open-source implementation of the gradient boosted tree algorithm. Gradient boosting is a supervised learning algorithm, which attempts to accurately predict a target variable by combining estimates from a set of simpler and weaker models [16].

For the present work, Python version 3.8 was used as the programming language of choice for running machine learning algorithms. Anaconda is the Python distribution used; it is delivered with all the tools and libraries needed to do machine learning, such as Numpy, Matplotlib, sklearn, Jupiter, Spider...etc.

2.4. Data Collection and Preparation

- **Data collection**

Recall that the purpose of this study is to contribute to the creation of an effective fraud detection model for a telecommunication network in order to reduce or eliminate losses caused by fraud. Therefore, we need to develop a model that can identify each fraudster and stop his activities. In order to do this, we started by collecting and processing data. As a result, we obtained CDR files with 60,000 call lines that we sorted then selected the fields we needed to build our model. We were granted a special permission to use this data while preserving the confidentiality of the user's information.

- **Data preparation**

It is our responsibility to understand, analyze and determine what data can be used to build our model.

- **Description of the data**

The CDRs data we collected from the MSOFTX3000 are in .csv format.

The CDRs from the MSOFTX3000 are dated APRIL 2021. The following Table 1 lists the different fields present:

Table 1. Overview of MSOFTX3000 CDR fields.

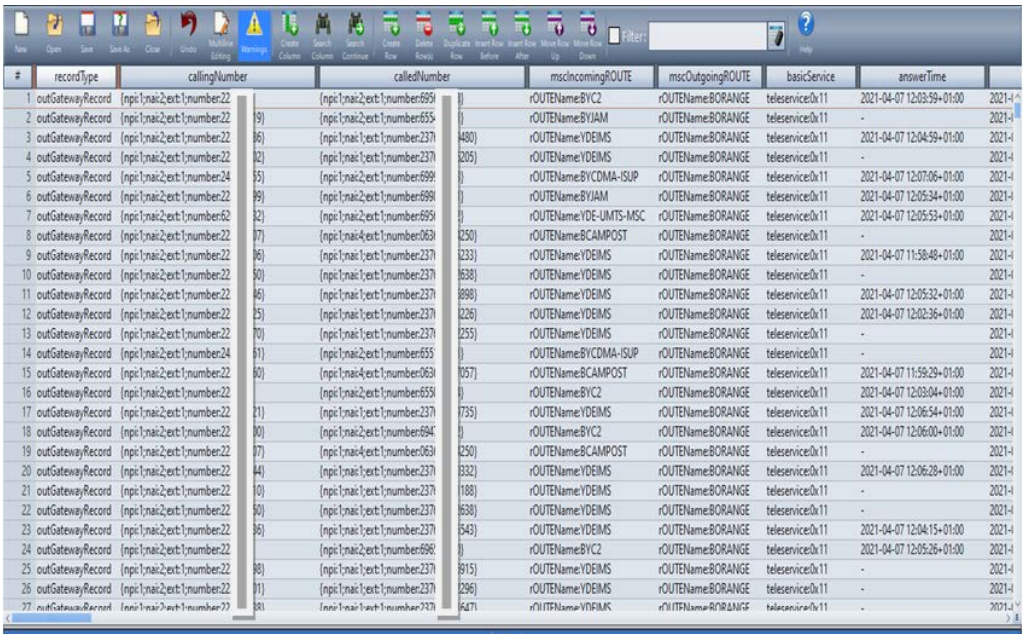
Element	Description
Calling number	The phone number of a caller
Called number	The phone number of a called party
Answer Date	The date the call was picked up
Answer time	The time when the call was picked up
Release date	Date the call was hung up
Release time	Time of call hang up
Call duration	Duration of the call.
Cause for term	Cause of call break

- **Exploring the Data**

It is important to visualize the data as it was collected and to show how the different domains relate to each other. The choice of datasets to be manipulated is crucial. Below is an image of some of the data fields.

- **Investigations of fraudulent numbers**

The fraudulent SIM Box accounts were investigated by the operator's fraud department and cancelled due to their malicious activity. As a result, we have obtained data tagged for the month of APRIL 2021; this presented by Figure 4 and Figure 5 presents a sample of data collected from the HUAWEI MSOFTX3000 which is the core network switching equipment.



#	recordType	callingNumber	calledNumber	mscIncomingROUTE	mscOutgoingROUTE	basicService	answerTime
1	outGatewayRecord	{npi:1;nai:2;ext:1;number:22	{npi:1;nai:2;ext:1;number:699	rOUTEName:BYC2	rOUTEName:BORANGE	teleservice:0x11	2021-04-07 12:03:59+01:00
2	outGatewayRecord	{npi:1;nai:2;ext:1;number:22	{npi:1;nai:2;ext:1;number:655	rOUTEName:BYJAM	rOUTEName:BORANGE	teleservice:0x11	-
3	outGatewayRecord	{npi:1;nai:2;ext:1;number:22	{npi:1;nai:1;ext:1;number:237	rOUTEName:YDEIMS	rOUTEName:BORANGE	teleservice:0x11	2021-04-07 12:04:59+01:00
4	outGatewayRecord	{npi:1;nai:2;ext:1;number:22	{npi:1;nai:2;ext:1;number:237	rOUTEName:YDEIMS	rOUTEName:BORANGE	teleservice:0x11	-
5	outGatewayRecord	{npi:1;nai:2;ext:1;number:24	{npi:1;nai:2;ext:1;number:699	rOUTEName:BYCDMA-ISUP	rOUTEName:BORANGE	teleservice:0x11	2021-04-07 12:07:06+01:00
6	outGatewayRecord	{npi:1;nai:2;ext:1;number:22	{npi:1;nai:2;ext:1;number:699	rOUTEName:BYJAM	rOUTEName:BORANGE	teleservice:0x11	2021-04-07 12:05:34+01:00
7	outGatewayRecord	{npi:1;nai:2;ext:1;number:62	{npi:1;nai:2;ext:1;number:699	rOUTEName:YDE-UMTS-MSC	rOUTEName:BORANGE	teleservice:0x11	2021-04-07 12:05:53+01:00
8	outGatewayRecord	{npi:1;nai:2;ext:1;number:22	{npi:1;nai:4;ext:1;number:063	rOUTEName:BCAMPOST	rOUTEName:BORANGE	teleservice:0x11	-
9	outGatewayRecord	{npi:1;nai:2;ext:1;number:22	{npi:1;nai:1;ext:1;number:237	rOUTEName:YDEIMS	rOUTEName:BORANGE	teleservice:0x11	2021-04-07 11:58:48+01:00
10	outGatewayRecord	{npi:1;nai:2;ext:1;number:22	{npi:1;nai:1;ext:1;number:237	rOUTEName:YDEIMS	rOUTEName:BORANGE	teleservice:0x11	-
11	outGatewayRecord	{npi:1;nai:2;ext:1;number:22	{npi:1;nai:1;ext:1;number:237	rOUTEName:YDEIMS	rOUTEName:BORANGE	teleservice:0x11	2021-04-07 12:05:32+01:00
12	outGatewayRecord	{npi:1;nai:2;ext:1;number:22	{npi:1;nai:1;ext:1;number:237	rOUTEName:YDEIMS	rOUTEName:BORANGE	teleservice:0x11	2021-04-07 12:02:36+01:00
13	outGatewayRecord	{npi:1;nai:2;ext:1;number:22	{npi:1;nai:1;ext:1;number:237	rOUTEName:YDEIMS	rOUTEName:BORANGE	teleservice:0x11	-
14	outGatewayRecord	{npi:1;nai:2;ext:1;number:24	{npi:1;nai:2;ext:1;number:655	rOUTEName:BYCDMA-ISUP	rOUTEName:BORANGE	teleservice:0x11	-
15	outGatewayRecord	{npi:1;nai:2;ext:1;number:22	{npi:1;nai:4;ext:1;number:063	rOUTEName:BCAMPOST	rOUTEName:BORANGE	teleservice:0x11	2021-04-07 11:59:29+01:00
16	outGatewayRecord	{npi:1;nai:2;ext:1;number:22	{npi:1;nai:2;ext:1;number:655	rOUTEName:BYC2	rOUTEName:BORANGE	teleservice:0x11	2021-04-07 12:03:04+01:00
17	outGatewayRecord	{npi:1;nai:2;ext:1;number:22	{npi:1;nai:1;ext:1;number:237	rOUTEName:YDEIMS	rOUTEName:BORANGE	teleservice:0x11	2021-04-07 12:06:54+01:00
18	outGatewayRecord	{npi:1;nai:2;ext:1;number:22	{npi:1;nai:2;ext:1;number:694	rOUTEName:BYC2	rOUTEName:BORANGE	teleservice:0x11	2021-04-07 12:06:00+01:00
19	outGatewayRecord	{npi:1;nai:2;ext:1;number:22	{npi:1;nai:4;ext:1;number:063	rOUTEName:BCAMPOST	rOUTEName:BORANGE	teleservice:0x11	-
20	outGatewayRecord	{npi:1;nai:2;ext:1;number:22	{npi:1;nai:1;ext:1;number:237	rOUTEName:YDEIMS	rOUTEName:BORANGE	teleservice:0x11	2021-04-07 12:06:28+01:00
21	outGatewayRecord	{npi:1;nai:2;ext:1;number:22	{npi:1;nai:1;ext:1;number:237	rOUTEName:YDEIMS	rOUTEName:BORANGE	teleservice:0x11	-
22	outGatewayRecord	{npi:1;nai:2;ext:1;number:22	{npi:1;nai:1;ext:1;number:237	rOUTEName:YDEIMS	rOUTEName:BORANGE	teleservice:0x11	-
23	outGatewayRecord	{npi:1;nai:2;ext:1;number:22	{npi:1;nai:1;ext:1;number:237	rOUTEName:YDEIMS	rOUTEName:BORANGE	teleservice:0x11	2021-04-07 12:04:15+01:00
24	outGatewayRecord	{npi:1;nai:2;ext:1;number:22	{npi:1;nai:2;ext:1;number:696	rOUTEName:BYC2	rOUTEName:BORANGE	teleservice:0x11	2021-04-07 12:05:26+01:00
25	outGatewayRecord	{npi:1;nai:2;ext:1;number:22	{npi:1;nai:1;ext:1;number:237	rOUTEName:YDEIMS	rOUTEName:BORANGE	teleservice:0x11	-
26	outGatewayRecord	{npi:1;nai:2;ext:1;number:22	{npi:1;nai:1;ext:1;number:237	rOUTEName:YDEIMS	rOUTEName:BORANGE	teleservice:0x11	-

Figure 4. HUAWEI MSOFTX3000 CDR Observations.

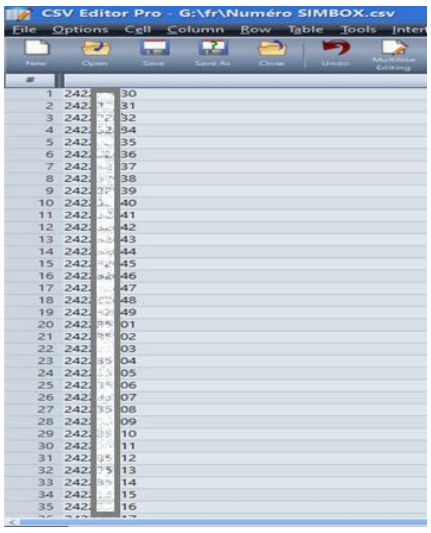


Figure 5. Sample of data from HUAWEI MSOFTX3000 CDR.

Note: *In* Figure 4 *and* Figure 5 *subscribers' phone numbers were blurred intentionally to protect their privacy.*

2.5. Evaluation Method

Confusion matrix

The confusion matrix is the commonly used method to describe and characterize the performance of the classification model in the fraud detection system. The confusion matrix is a kind of summary of the prediction results for a particular classification problem. It compares the actual data for a target variable to that predicted by a model. Right and wrong predictions are revealed and divided by class, allowing them to be compared with defined values. The results of a confusion matrix are classified into four broad categories: true positives, true negatives, false positives, and false negatives [17].

Different metrics can be calculated from the contingency Table 2 to facilitate interpretation. This is, as example, the case **for the error rate, Accuracy, precision, recall and F1 score**. These indicators allow a better appreciation of the quality of the model's precision.

2.6. Construction and Training of the Model

In this part, we built the columns materializing the volume of incoming and outgoing calls of each number and build the fraud target variable (binary variable worth 1 if the call is fraudulent and 0 otherwise), Figure 6 presents the construction of new columns.

- **Labelling:**

We have for the column "is_fraudulent" labelled the SIM Box numbers. Thus, on each line we apply the lambda function which searches in a line; if there is a fraudulent number; it returns 1 and if not it returns 0, this is represented in Table 3.

Table 2. Confusion matrix.

		Actual values	
		P	N
Predictions	P	TP	FP
	N	FN	TN

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 3510 entries, 0 to 5571
Data columns (total 8 columns):
 #   Column                  Non-Null Count
---  ------                  --------------
 0   callingNumber           3510 non-null
 1   normalRelease           3510 non-null
 2   partialRecord           3510 non-null
 3   unsuccessfulCallAttempt 3510 non-null
 4   outcoming_call_volume   3510 non-null
 5   mean_call_duration      3510 non-null
 6   incoming_call_volume    3510 non-null
 7   is_fraudulent           3510 non-null
```

Figure 6. Construction of new columns.

Table 3. Labelling format.

Cible	Code
Numéros non frauduleux (Not Fraud)	0
Numéros frauduleux (Fraud)	1

- **Data transformation**

We determined the outgoing call volume for each number:

Outcoming_call_volume: in our dataset, we select the numbers that appear several times, then we group them together by calling numbers and for each group we add the number of times it is in the dataset more precisely at the level of the calling number column.

We have determined the incoming call volume for each number:

Incoming_call_volume: We select the outgoing call number from the list of aggregated numbers and search for the number of times it appears in the list of called numbers within the initial dataset.

We determined the average call duration that a number had to make:

Mean_call_duration: represents the ratio between the total duration of calls by the total number of calls.

- **Data normalization**

Since the machine learning platform does not understand strings, we had to encode the classes of the cause for term variables using the one hot encoding method which is a very common approach An encoding creates new (binary) columns, indicating the presence of each possible value from the original data: "normalRelease" to represent normal hang-up, "partialRecord" to represent Partial record and "nsuccesfulCallAttempt" to represent Unsuccessful calls.

- **Search for dependency between variables**

The closer the value is to 1 (a solid red), the stronger and more positive is the correlation. On the other hand, if the correlation is close to 0 (dark blue), the correlation is very negative. This is presented by Figure 7.

- **Training the model**

Therefore, we used the normal dataset splitting rule found in the Python Pandas library for our dataset with 80% set for training and 20% for testing. The function used to split the data set into training data and test data is present below.

```
X_train, X_test, y_train, y_test = train_test_split(X_train_res, y_train_res, random_state = 40, test_size = 0.2)
```

3. Results and Discussion

3.1. Learning and Creating Prediction Models

- **Prediction with the Random Forest**

To do this, we imported the algorithm from the sklearn library via the following code:

```
from sklearn.ensemble import RandomForestClassifier
```

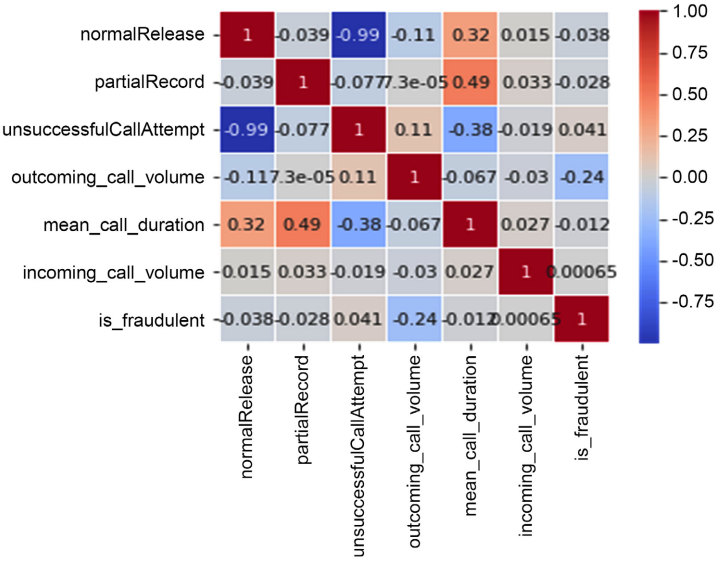


Figure 7. Correlation of data.

Then we created a Random forest classifier of 100 trees via the following code:

rf = RandomForestClassifier(n_estimators = 100, random_state = 40)

And we launched the training on our training dataset with the following python code:

rf.fit(X=X_train, y=y_train)

After training our Random Forest model, we obtained the following in **Figure 8**.

We had an accuracy to determine the fraudsters of 0.86 with an f1-score of 0.94 and an accuracy of the non-fraudsters of 0.95 with an f1-score of 0.88, and a total accuracy of 0.91 so our model predicted well in training.

After testing the **Random Forest model**, we obtained the result presented in **Figure 9**.

For the test, on the one hand we had an accuracy to determine the fraudsters of 0.88 with an f1-score of 0.96 and on the other hand we had an accuracy to determine the non-fraudsters of 0.96 with an f1-score of 0.89, and the trained model had a general accuracy of 0.92 so our model reacted well to the data test.

- **Prediction with the SVM**

To do this, we imported the algorithm from the sklearn library via the following code:

from sklearn.svm import SVC

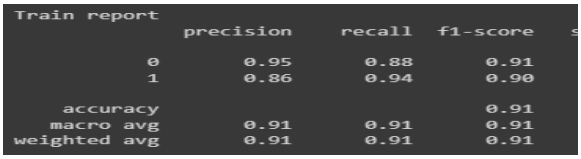
Then we created an SVM whose C value determines the penalty for the classifier. Presented via the following codes:

svc = SVC(random_state = 40, C = 20)

And we launched the training on our training dataset with the following python code:

svc.fit(X = X_train, y = y_train)

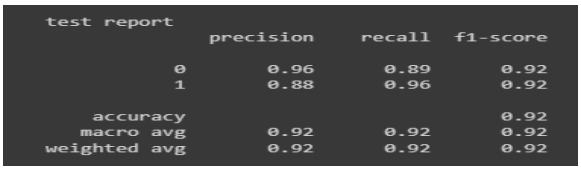
Training our SVM model, we obtained the following result presented in **Figure 10**.



Train report

	precision	recall	f1-score	s
0	0.95	0.88	0.91	
1	0.86	0.94	0.90	
accuracy			0.91	
macro avg	0.91	0.91	0.91	
weighted avg	0.91	0.91	0.91	

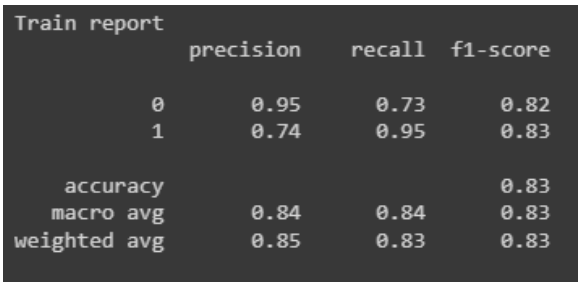
Figure 8. Train random forest.



test report

	precision	recall	f1-score
0	0.96	0.89	0.92
1	0.88	0.96	0.92
accuracy			0.92
macro avg	0.92	0.92	0.92
weighted avg	0.92	0.92	0.92

Figure 9. Random forest test.



Train report

	precision	recall	f1-score
0	0.95	0.73	0.82
1	0.74	0.95	0.83
accuracy			0.83
macro avg	0.84	0.84	0.83
weighted avg	0.85	0.83	0.83

Figure 10. SVM train.

We had an accuracy to determine fraudsters of 0.74 with an f1-score of 0.83 and an accuracy of non-fraudsters of 0.95 with an f1-score of 0.83, and a general accuracy of 0.83. Here our model performed at a lower accuracy for the detection of fraudsters and non-fraudsters in training.

Testing our **SVM model,** we obtained the result in **Figure 11**:

For the test, on the one hand we had an accuracy to determine the cheaters of 0.89 with an f1-score of 0.53 and on the other hand an accuracy of the non cheaters of 0.64 with an f1-score of 0.53, and the trained model had a general an accuracy of 0.69 so our model did not react well to the data test.

- **Prediction with the XGBoost**

To do this, we imported the algorithm from the **sklearn** library via the following code:

from xgboost import XGBClassifier

Then we created a GaussianNB via the following code:

nb = GaussianNB()

And we launched the training on our training dataset with the following python code:

nb.fit(X = X_train, y = y_train)

Training our XGBoost model, we got the following result in **Figure 12**:

We had accuracy for determining fraudsters of 0.71 with an f1-score of 0.82 and accuracy for non-fraudsters of 0.96 with an f1-score of 0.80, and a general accuracy of 0.81 so our model predicted well in training.

Testing **XGBoost** model, we obtained the results presented in **Figure 13**.

test report	precision	recall	f1-score
0	0.64	0.96	0.76
1	0.89	0.37	0.53
accuracy			0.69
macro avg	0.76	0.67	0.65
weighted avg	0.75	0.69	0.65

Figure 11. SVM test.

Train report	precision	recall	f1-score
0	0.96	0.68	0.80
1	0.71	0.97	0.82
accuracy			0.81
macro avg	0.84	0.82	0.81
weighted avg	0.85	0.81	0.81

Figure 12. XGBoost train.

test report	precision	recall	f1-score
0	0.96	0.67	0.79
1	0.72	0.97	0.83
accuracy			0.81
macro avg	0.84	0.82	0.81
weighted avg	0.85	0.81	0.81

Figure 13. XGBoost test.

For the testing, on the one hand we had an accuracy to determine the fraudsters of 0.72 with an f1-score of 0.83 and on the other hand an accuracy of the non-fraudsters of 0.72 with an f1-score of 0.79, and the trained model had a general accuracy of 0.81 so our model had an acceptable reaction to the data test.

3.2. Evaluation of the Model by the Confusion Matrix

- **Random Forest algorithm**

Figure 14 presents Random Forest confusion matrix, in this confusion matrix, the number of false negatives is 23 so we predicted **"no"** but they are fraudsters while the number of false positives is 66 we predicted **"yes"** but they are not fraudsters. The number of true positives is 519, thus we predicted that they are not fraudsters and indeed they are not fraudsters, and the number of true negatives is 492 so we predicted that they are fraudsters and indeed they are fraudsters.

- **SVM algorithm**

Figure 15 presents SVM confusion matrix, in this confusion matrix, the number of false negatives is 322 so we predicted that they are not fraudsters but they are fraudsters, the number of false positives is 24 we predicted **"yes"** but they are not fraudsters, the number of true positives is 561 we predicted that

they are not fraudsters and indeed they are not fraudsters, and the number of true negatives is 193 we predicted that they are fraudsters and indeed they are fraudsters.

- **XGBoost algorithm**

Figure 16 presents XGBoost confusion matrix, in this confusion matrix, the number of false negatives is 16 so we predicted that they are not fraudsters but they are fraudsters, the number of false positives is 192 we predicted yes but they are not fraudsters, the number of true positives is 393 we predicted that they are not fraudsters and indeed they are not fraudsters, and the number of true negatives is 499 we predicted that they are fraudsters and indeed they are fraudsters

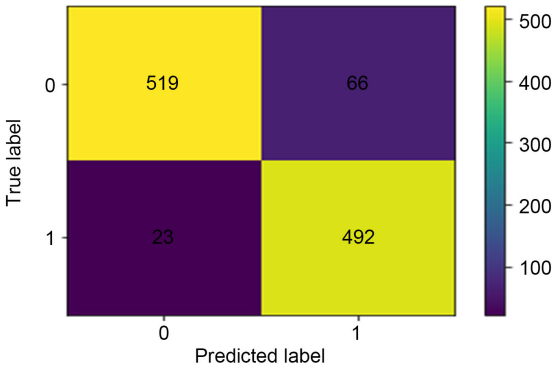


Figure 14. Random Forest confusion matrix.

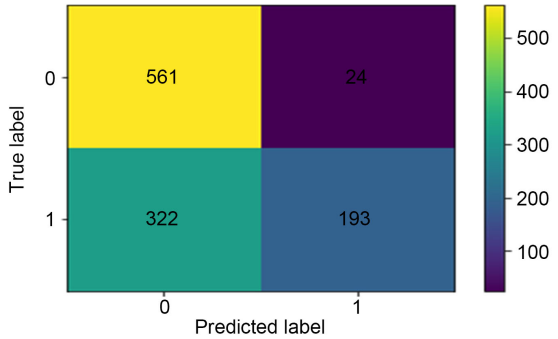


Figure 15. SVM confusion matrix.

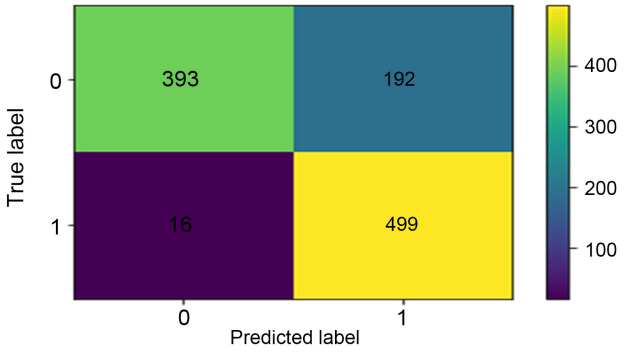


Figure 16. XGBoost confusion matrix.

3.3. Discussion

As a follow-up to the experimental research we have done in the previous paragraphs, the machine learning model we propose is the Random Forest model. Indeed, this model is retained because it predicts the optimal SIM Box fraud detection solution with an accuracy of 0.92 and a score of 0.92, as presented in Table 4:

We made the prediction with our best performing model, and determine if Random Forest model is able to correctly determine a case of SIM Box fraud. Figure 17 presents the command which can be used.

Then we tested each line of the dataset to bring out the fraudulent and non-fraudulent numbers. We obtained the dataset with the list of fraudulent and non-fraudulent numbers.

Finally we tested each line of our dataset to highlight the only fraudulent numbers without the lines of the dataset; we obtained the dataset with the list of fraudulent numbers. For that we used the following code and the figures presenting that results are Figure 18 and Figure 19:

Table 4. Comparison of the models.

		Precision	Accuracy	F1-score
Random Forest	0	0.96	0.92	0.92
	1	0.88		0.92
SVM	0	0.64	0.69	0.76
	1	0.89		0.53
XGBoost	0	0.96	0.81	0.79
	1	0.71		0.83

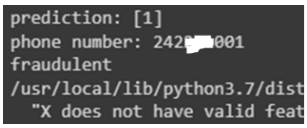


Figure 17. Fraud case prediction.

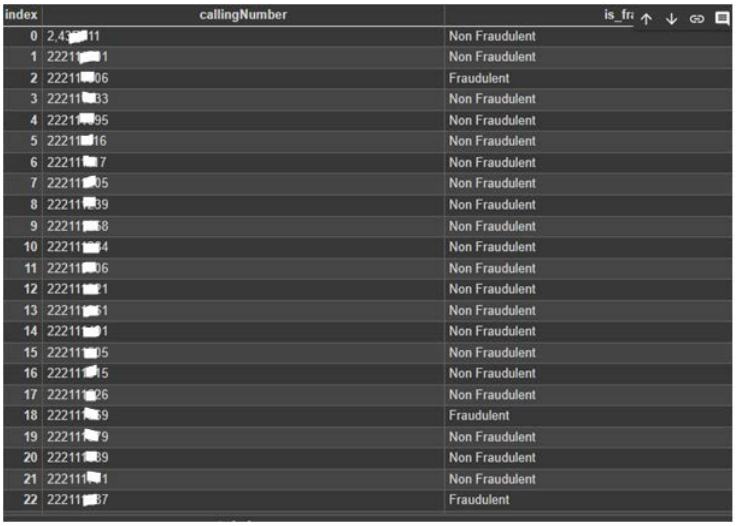
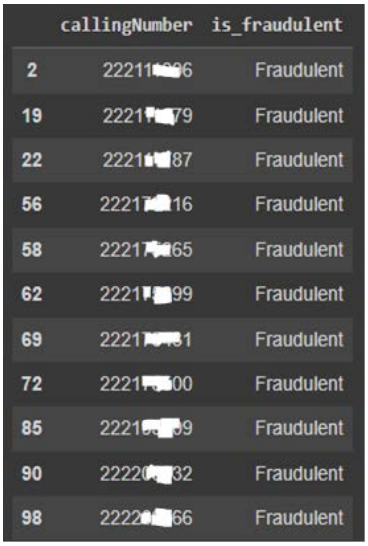


Figure 18. Detected targets.



	callingNumber	is_fraudulent
2	22211[redacted]6	Fraudulent
19	2221[redacted]79	Fraudulent
22	2221[redacted]87	Fraudulent
56	2221[redacted]16	Fraudulent
58	2221[redacted]65	Fraudulent
62	2221[redacted]99	Fraudulent
69	2221[redacted]1	Fraudulent
72	2221[redacted]00	Fraudulent
85	2221[redacted]9	Fraudulent
90	2222[redacted]32	Fraudulent
98	2222[redacted]66	Fraudulent

Figure 19. Fraudulent numbers detected.

Df_fraudulents = dataframe_predictions_with_numbers[dataframe_ predictions_with_numbers["is_fraudulent"] = = ["fraudulent"]

Due to the rapid evolution of the SIM Box fraud, we think that it is necessary to refresh the detection model periodically, like every quarter and always use the more accurate model for fraud detection.

4. Conclusions

The objective of this paper consists of researching and implementing a SIM Box fraud detection system for a telecommunications network operator, with a case study based on data collected to a fixed and mobile network operator in Cameroon. The project aims to quickly identify SIM Box fraud and reduce or eliminate the financial loss caused by the scam in the company's turnover.

We used machine learning techniques to effectively identify SIMboxing fraud based on CDR analysis and prevent it from harming telecom companies in terms of revenue, quality of service and security. In order to detect the SIM Box scam, since the dataset is unbalanced, we used classification algorithms. After this step, we performed a comparison of the incoming and outgoing call rates, and then we determined the total duration of a call in a day. Thus, an individual not detected during the first hours may be detected in the following hours. We ran the data under different Machine Learning models of unsupervised learning in order to compare the performance of different models based on their accuracy and select the best one for fraud detection. From the experiment, we found that Random Forest, SVM and XGBoost are able to detect the bypass SIM box fraud. The experimental results showed that Random Forest has the best accuracy compared to the others. Random Forest gave 92% accuracy while SVM model gave 76% accuracy and XGBoost gave 84% accuracy. Therefore, the Random Forest approach is more suitable for the classification model used for SIM BOX fraud detection with 92% accuracy. Then this model has been used to identify the

fraudulent numbers in the mobile operator's network successfully.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Agence Ecofin (2015) Cameroun: 22,2 milliards FCfa de pertes en 2015 sur les appels téléphoniques frauduleux par Simbox. Agence Ecofin. https://www.agenceecofin.com/gestion-publique/0910-32980-cameroun-22-2-milliards-fcfa-de-pertes-en-2015-sur-les-appels-telephoniques-frauduleux-par-simbox

[2] Cameroun: La fraude par simbox peut coûter 18 milliards de F CFA aux opérateurs. JeuneAfrique. https://www.jeuneafrique.com/271158/economie/cameroun-la-fraude-par-simbox-peut-couter-18-milliards-de-f-cfa-aux-operateurs/

[3] Revenue Assurance and Fraud: Risque et fraud management (revenue assurance). http://risquefraudtelecom.blogspot.com/2012/05/introduction-au-risque-et-fraud.html

[4] Karunathilaka, A.V.V.S. (2020) Fraud Detection on International Direct Dial Calls. University of Colombo School of Computing, Colombo. https://dl.ucsc.cmb.ac.lk/jspui/bitstream/123456789/4451/1/2016%20MIT%20029.pdf

[5] Reaves, B., Shernan, E., Bates, A., Carter, H. and Traynor, P. (2015) Boxed Out: Blocking Cellular Interconnect Bypass Fraud at the Network Edge. *Proceedings of the* 24*th USENIX Security Symposium*, Washington DC, 12-14 August 2015. https://www.usenix.org/system/files/conference/usenixsecurity15/sec15-paper-reaves-boxed.pdf

[6] Witten, I.H., Frank, E. and Hall, M.A. (2016) Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufmann, Burlington.

[7] Ighneiwa, I. and Mohamed, H.S. (2017) Bypass Fraud Detection: Artificial Intelligence Approach. https://www.researchgate.net/publication/321070233_Bypass_Fraud_Detection_Artificial_Intelligence_Approach

[8] Krenker, A., Volk, M., Sedlar, U., Bešter, J. and Kos, A. (2009) Bidirectional Artificial Neural Networks for Mobile-Phone Fraud Detection. *ETRI Journal*, **31**, 92-94. https://doi.org/10.4218/etrij.09.0208.0245

[9] Sallehuddin, R., Ibrahim, S., Mohd Zain, A. and Hussein Elmi, A. (2015) Detecting SIM Box Fraud by Using Support Vector Machine and Artificial Neural Network. *Jurnal Teknologi*, **74**, 131–143. https://doi.org/10.11113/jt.v74.2649

[10] Deussom Djomadji, E.M., Matemtsap Mbou, B., Tchagna Kouanou, A., Ekonde Sone, M. and Bayonbog, P. (2022) Machine Learning-Based Approach for Designing and Implementing a collaborative Fraud Detection Model through CDR and Traffic Analysis. *Transactions on Engineering and Computing Sciences*, **10**, 46-58. https://doi.org/10.14738/tmlai.104.12854

[11] Subudhi, S. and Panigrahi, S. (2017) Use of Possibilistic Fuzzy C-Means Clustering for Telecom Fraud Detection. In: Behera, H., Mohapatra, D.., Eds., *Computational Intelligence in Data Mining. Advances in Intelligent Systems and Computing*, Vol. 556, Springer, Singapore, 633-641. https://doi.org/10.1007/978-981-10-3874-7_60

[12] Bernabe, B., Michel, D., Marie, C. and Fabrice, M. (2022) Comparing Machine Learning Algorithms for Improving the Maintenance of LTE Networks Based on Alarms Analysis. *Journal of Computer and Communications*, **10**, 125-137. https://doi.org/10.4236/jcc.2022.1012010

[13] Random Forest. Data Analytics Post. https://dataanalyticspost.com/Lexique/random-forest/

[14] Schott, M. (2019) Random Forest Algorithm for Machine Learning. Capital One Tech. https://medium.com/capital-one-tech/random-forest-algorithm-for-machine-learning-c4b2c8cc9feb

[15] Support Vector Machine. Wikipedia. https://fr.wikipedia.org/wiki/Machine_%C3%A0_vecteurs_de_support

[16] XGBoost. Machine Learning Book. https://vatsalparsaniya.github.io/ML_Knowledge/XGBoost/Readme.html

[17] Matrice de confusion: Comment la lire et l'interpréter? https://www.jedha.co/formation-ia/matrice-confusion