khazanah informatika

Jurnal Ilmu Komputer dan Informatika

Performance Analysis of Isolation Forest Algorithm in Fraud Detection of Credit Card Transactions

Indra Waspada*, Nurdin Bahtiar, Panji Wisnu Wirawan, Bagus Dwi Ari Awan
Department of Computer Science
Universitas Diponegoro
Semarang, Indonesia
*correspondence: indrawaspada@lecturer.undip.ac.id

Abstract-Losses incurred due to fraud on e-commerce transactions, especially those based on credit cards, continue to increase, resulting in large losses each year. One mechanism to minimize the risk of fraudulent credit card transactions is to utilize a detection technique for ongoing transactions. Credit card transaction data in its original state does not have a label, and the amount of fraud data on the training data is very small so that it belongs to a very unbalanced category, and the pattern of fraud continues to change. Isolation forest is an unsupervised algorithm that is efficient in detecting anomalies. Several techniques can be applied to improve the performance of the Isolation forest model. Previous studies used the ROC-AUC metric in analyzing the performance of Isolation Forests, which could provide incorrect information. This study made two contributions; the first is to present a performance analysis with both the ROC-AUC and AUCPR. Thus, it can be seen that the high ROC-AUC value does not guarantee the model has the reliability in detecting fraud. In comparison, the information provided through AUCPR is more appropriate to describe the ability of the model to capture data fraud. The second contribution is to propose several techniques that can be applied to improve the performance of the Isolation forest model, namely to optimize the determination of the amount of training data, feature selection, the amount of fraud contamination, and setting hyper-parameters in the modeling stage (training). Experiments were carried out using a real-life dataset from ULB. The best results are obtained when the validation data split ratio is 60:40, using the five most important features, using only 60% of fraud data, and setting hyper-parameters with the number of trees 100, 128 sample maximum, and 0.001 contamination. The validation performance of this model is precision 0.809917, recall 0.710145, f1-score 0.756757, ROC-AUC 0.969728, and AUCPR 0.637993, while for Testing results obtained precision 0.807143, recall 0.763514, f1-score 0.784722, ROC-AUC 0.97371, and AUCPR 0.759228.

Keywords: credit card, fraud, Isolation forest, unsupervised, precision, recall, ROC-AUC, AUCPR

Article info: submitted March 10, 2020, accepted April 24, 2020

1. Introduction

The definition of fraud refers to the Black's Law Dictionary is "an act with attempts at fraud or violation by one or more individuals who are generally for financial gain" [1]. According to a report from the Association of Certified Fraud Examiners (ACFE) in 2018, there were losses of more than US $ 7 billion [2]. One category of fraud is credit card fraud. Fraud trends in transactions continue to increase, resulting in large money losses every year. It is estimated that losses increase each year at double-digit rates by 2020 [3]. This is because physical cards are not needed in an online transaction environment, and information from the card is enough to complete the payments. This makes it easier to commit fraud than before.

One mechanism to minimize the risk of credit card fraud is to use a detection technique for ongoing transactions to identify potential fraud. Several machine learning and data mining techniques have been used in research into credit card fraud detection. The research included supervised learning based on neural networks [4], [5], rule base [4], logistic regression [6], [7], SVM [5] - [7], Random Forest [6] - [9], based on semi-supervised learning using Graph [8], and balanced Random Forest [10], and based on unsupervised-learning using auto-encoder [11], clustering [12], self organizing map [13], local outlier factor [14].

With the size of the current transaction data that is very large then to do a credit card fraud analysis must pay attention to aspects of the study of big data [15]. Some assumptions need to be considered in the condition of

a very large transaction data flow (big data), including that the data obtained are generally without labels (fraud or normal) and the number of incidents of fraud is very small compared to normal (skew). This assumption is of concern in several anomalous detection studies with one of the conclusions that the unsupervised learning algorithm excels at handling predictive tasks for large amounts of unbalanced data [16], [17]. The three unsupervised algorithms most commonly used are local outlier factor (LOF), one-class SVM (OCSVM), and Isolation forest (iForest) [16], [18], [19]. Of the three, iForest is the most efficient in detecting anomalies because it has good scalability capabilities for big data and reasonable memory usage for large sample sizes, so it is very suitable for the production environment [20].

Credit card transaction data in its original state does not have a label, other than that the amount of fraud data on the training data is very small so that it includes a very unbalanced category (highly imbalanced data) as well as a new pattern of fraud that is very open. In this study, a dataset from Kaggle originating from a credit card company in Europe was used. Some previous studies using this data set [21] - [23].

The study of Ounacer et al. [21] showed that Isolation forest is superior compared to an unsupervised algorithm by comparing performance using f1-score metrics, accuracy, and ROC-AUC (Receiver Operator Characteristic - Area Under Curve). Niu et al. [22] compared six supervised models with four unsupervised models. However, the supervised model is balancing using the normal data undersampling technique from 284,807 to 492 (0.17%) so that the resulting model is very vulnerable to bias and overfitting. The criticism for the ROC-AUC metric used by Ounacer et al. [21] and Niu et al. [22] is that the use of ROC-AUC for highly unbalanced data types can provide inappropriate information as a more appropriate alternative is the AUCPR metric (Area Under Curve - Precision-Recall) [24], [25].

The first contribution made in this study is to present a performance analysis with both the ROC-AUC and AUCPR to prove the best metrics for providing appropriate information. Second, building the Isolation forest model in detecting credit card transaction fraud through performance analysis by performing some optimization on the determination of the amount of training data, feature selection, the amount of fraud contamination, and setting hyper-parameters in the modeling stage so that the highest performing model can be obtained both validation data and Testing data shown through metric precision, recall, f1-score, ROCAUC, and AUCPR.

2. Method

This section discusses the theory of the Isolation forest algorithm and continues with the stages of the research scenario starting from understanding data, feature selection, modeling, and evaluation.

a. Isolation forest

Isolation forest is an unsupervised learning method formed from a collection of isolation trees or iTrees from a given data set, where the data will be called an anomaly when it has the shortest average path length on the iTree.

Anomaly detection with iForest is done in two stages. The first is the training phase (training) to build an isolation tree using subsamples from the training data set. The second stage (testing) provides the test instance to the isolation tree to get an anomaly value for each instance.

1. Training Stage

   At this stage, iTrees are created by recursively partitioning the training data set until all instances are isolated or when the maximum height of the tree is reached, which results in a partial model. The height limit of tree l is automatically set by the size of the sub-sample ψ: $l=ceiling(log_2 \psi)$, which estimates the average height of the tree. The training stage algorithm can be seen in algorithms 1 and 2.

Algorithm 1 : *iForest(X,t,ψ)*

Inputs: *X* - input data, *t* - number of *trees*, ψ - sub-sampling size

Outputs: a set of *t iTrees*

Initialize *Forest*
set *height limit l=ceiling(log₂ψ)*
for *i=1* **to** *t* **do**
X'←sample(X, ψ)
Forest←Forest ∪ iTree(X',0,l)
end for
return *Forest*

Algorithm 2 : *iForest(X,t,ψ)*

Inputs: *X* - input data, *e* - current *tree* height, *l* – height limit

Outputs: an *iTree*

if $e \geq l$ *or* $|X| \leq 1$ **then**
return *exNode{*Size←$|X|$*}*
else
let *Q* be a list attributes in *X*
randomly select an attribute q ∈ Q
randomly select a split point p from max and min values of attribute *q* in *X*
$X_l \leftarrow$ *filter* $(X, q < p)$
$X_r \leftarrow$ *filter*$(X, q \geq p)$
return *inNode{ Left* ← *iTree*$(X_l, e + 1, l)$,
Right←iTree(X_r,e+1,l),
SplitAtt← q,
SplitValue←p}
end if

2. Evaluation Stage

   At the evaluation stage, the anomaly value s is obtained from the expected length of the E (h (x)) path of each test instant. E (h (x)) is obtained by passing instances through each iTree in iForest. By using the PathLength function, the length of the single path h (x) is obtained by counting the number of ends e from the root node to the termination node that is passed by the x instance in iTree. When x is terminated on an external node, with *Size* > 1, the value of e and setting c (*Size*) is obtained. When all h (x) has been obtained, the anomaly value can be calculated with formula 2. Detailed PathLength function can be seen in algorithm 3

Algorithm 3: *PathLength (x, T, e)*

Inputs: x – an instance, T – an i*Tree*, e – current path length; to be initialized to zero when first called

Outputs: path length of x

1: **if** T is an external node **then**
2: **return** $e + c(T.size)$
3: **end if**
4: a ← $T.splitAtt$
5: **if** $x_a < T.splitValue$ **then**
6: **return** $PathLength\ (x, T.left, e + 1)$
7: **else** $\{x_a \geq T.splitValue\}$
8: **return** $PathLength(x, T.right, e + 1)$
9: **end if**

Because iTree has a structure equivalent to a Binary Search Tree, the estimated h (x) for terminating an external node is the same as the unsuccessful search on BST. By borrowing analysis from BST to estimate the average length of the iTree track. So with n instance data sets, the average length of the unsuccessful search path in BST can be obtained:

$$c(n) = 2H(n - 1) - (2(n - 1)/n)$$

where H (i) is a harmonic number and can be estimated with ln (i) + 0.5772156649 (Euler constant). Because c (n) is the average of h (x) for n, it can be used for normalization of h (x). The anomaly value s of instant x can be defined as:

$$s(x, n) = 2^{-E(h(x))/c(n)}$$

where E (h (x)) is the average value of h (x) of a group of isolation trees. In formula 2:

- When E(h(x))→ c(n),s→ 0.5;
- When E(h(x))→ 0,s→1;
- and When E(h(x))→n-1,s→0.

s monotonic of h (x). The relation between E (h (x)) and s satisfies the condition when 0 <s≤1 to 0 <h (x) ≤n-1. By using the anomaly value s, it can be determined:

(a) if the result of the instance s is very close to 1, then it is confirmed as an anomaly
(b) if the instant has s very small compared to 0.5, then it is very safe to be categorized as a normal instant, and
(c) if all instants produce s≈0.5 then all samples do not have anomalies.

b. Use of Data Sets

The data used in this study is a dataset obtained from Kaggle ULB. The dataset used is a data set containing credit card transactions conducted in September 2013 by European credit card companies for two days. This ULB dataset presents 284,807 credit card transactions, and 492 of them are fraudulent transactions. The number of fraudulent transactions, which is only 0.172% of the number of existing transactions, makes this ULB dataset very unbalanced. The ULB dataset contains numeric data with 30 feature columns and 1 label/class column. Original feature information from 28 features cannot be displayed and explained in detail because of confidentiality issues (V1, V2, ... V28). The main components of the 28 features are obtained by PCA. Two other features that have not been transformed by PCA are 'Time' and 'Amount'. The 'Time' feature contains the time that passes for each transaction in seconds. The 'Amount' feature contains the transaction fee amount. And the 'Class' column is the response variable and a value of 1 if fraud occurs and 0 if normal.

c. Feature Selection

In this study, feature selection uses a random forest algorithm and filter method. The random forest algorithm is used to calculate the importance score of each feature. Features with a small importance score will be filtered and not used. The feature set that will be used is the 10 best features based on the highest importance score. Figure 1. shows the feature selection process carried out.

d. Distribution of Data Sets

Modeling is done by dividing the dataset into two namely training data (70%) and Testing data (30%). Training data is used to create a model through several variations of the experimental scenario. The schema of the experiments carried out can be presented in Figure 2.

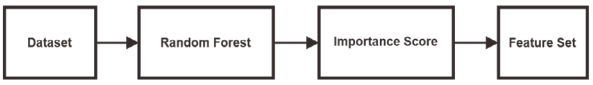

Figure 1. Feature selection process

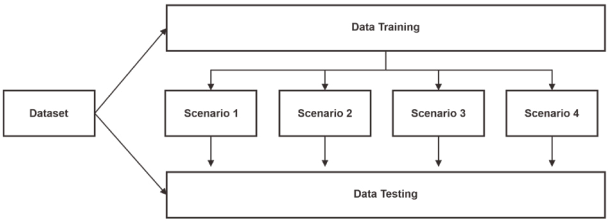

Figure 2. Experimental scenario

Broadly speaking, the experiment was conducted with four scenarios. Scenario 1 focuses on the split ratio used, scenario 2 focuses on the number of best features used, scenario 3 focuses on the ratio of fraud classes used in training, and scenario 4 focuses on the hyper-parameter settings used to create the model. for more details will be explained in detail in each scenario as follows :

1. In scenario 1, the training data is split into two parts, namely training model data and validation data with a variety of ratios. Model training data is used to create a classification model, and validation data is used to test the model created. The split ratios used are 90:10, 80:20, 70:30, and 60:40. The split ratio that produces the best model will be used in the next scenario. Scenario 1 can be seen in Figure 3.

2. In scenario 2 a feature selection is performed on the training data and split the data into two with the best ratio of the results of scenario 1. Feature selection is done that is using several variations of the best number of features. The variation in the number of features used is 1 to 10 of the best features. The number of features used that produce the best model will be used in the next scenario. Schema experiment scenario 2 can be seen in Figure 4.
3. In scenario 3 a feature selection is performed with the best number of features from scenario 2 and divides training data into two with the best split ratio of scenario 1. After that, the training model data is separated between data labeled a fraud and normal. The classification model will be made by regulating the ratio of data labeled a fraud. The ratio used is 100%, 90%, 80%, ..., 20%, 10%, 5% of the total amount of data labeled fraud in the training data model. Schema scenario 4 can be seen in Figure 5.
4. In scenario 4, hyper-parameter settings are performed on the best model produced in scenario 3. The parameters of the Isolation forest algorithm are regulated, namely the number of trees, maximum sample, and contamination. The parameters of the number of trees used are 10, 25, 50, 100, 150, and 200. The maximum parameters used are 64, 128, 256, 512, and 1024. Contamination parameters used are 0.0004, 0.0008, 0.001, 0.0015, 0.002. Schema scenario 4 can be seen in Figure 6.

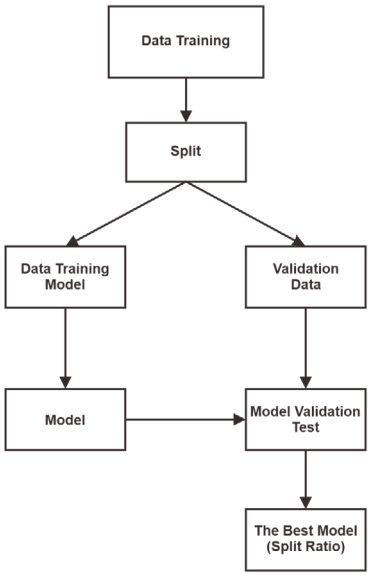

Figure 3. Schema scenario 1

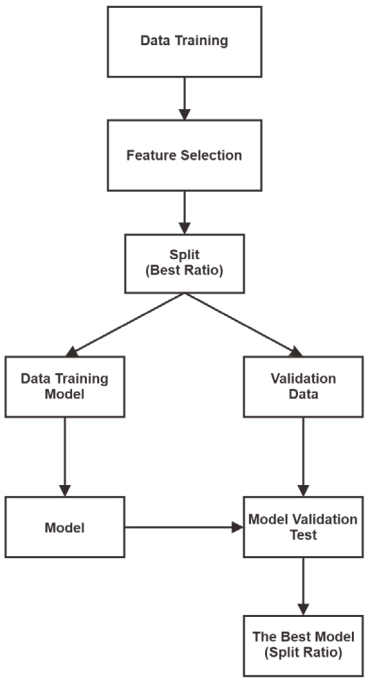

Figure 4. Schematic of scenario 2

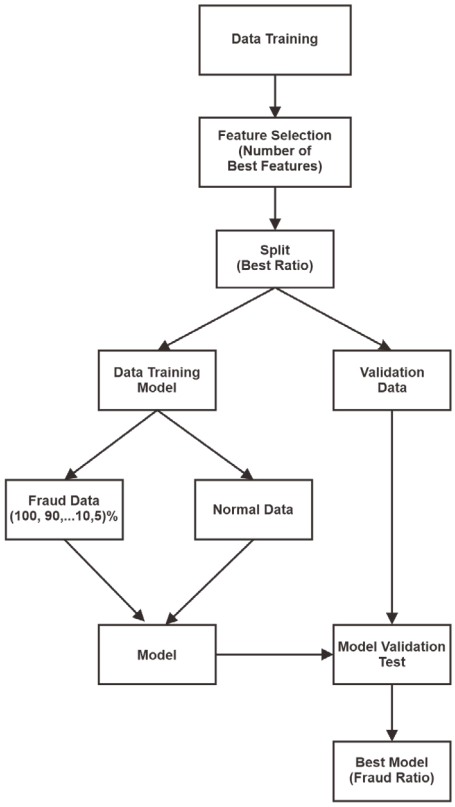

Figure 5. Scenario 3

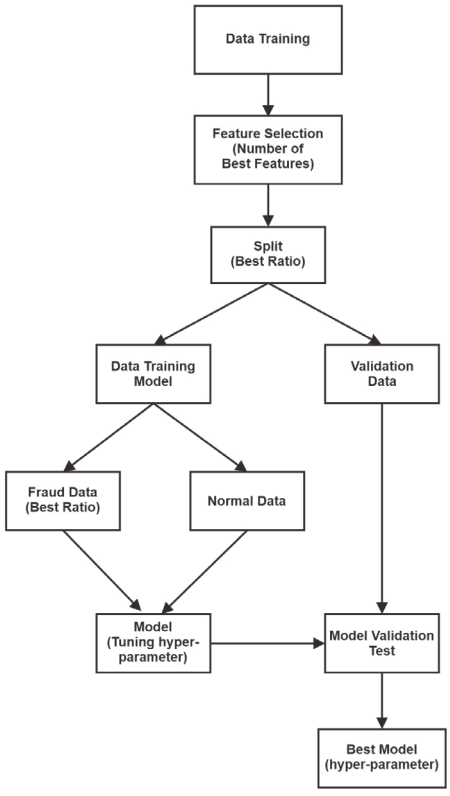

Figure 6. Schematic of scenario 4

e. Evaluation

Evaluation is carried out on training data for each experimental scenario (modeling) using Testing data. Classification model performance was calculated using precision, recall, f1-score, roc auc curve, and precision-recall curve.

3. Result

The experiment was documented in the Github repository (https://github.com/BagusDAriAwan/fraud). At the data understanding stage, no missing or blank data was found. While the results for the next stage, which are the results of feature selection and the results of each experimental scenario, are described in the following sub-chapters.

a. Results of Feature Selection Techniques

The results of feature selection using the random forest algorithm can be seen in Table 1. The information in Table 1. describes the attribute with a higher importance score, which has a stronger influence in classifying the data.

b. Experiment Scenario Results 1

Based on the evaluation results on the validation data, the best model with the training data ratio is 60%, and the validation data ratio is 40%. Detailed evaluation results on the validation data for each variation of the split data ratio can be seen in Table 2. The results of the evaluation of the best model using Testing data can be seen in Table 4. Figure 7 shows the best ROC curve and precision Recall model of the validation data, and in Figure 8 is the curve of the best model for Testing Data.

Table 1. Results of the random forest algorithm

Feature	Score	Feature	Score	Feature	Score
V14	0.191396	V19	0.012698	V23	0.006476
V4	0.156735	V8	0.011657	V22	0.006006
V17	0.154208	V6	0.010132	V1	0.005765
V12	0.153089	V27	0.009422	V24	0.005645
V11	0.075608	V3	0.007821	V28	0.005104
V2	0.051632	V26	0.007501	V18	0.004795
V10	0.038917	V9	0.007076	V16	0.004732
V7	0.015483	Amount	0.007075	V25	0.004156
V20	0.014762	V13	0.006597	V5	0.003216
V21	0.013873	V15	0.006509	Time	0.001916

Table 2. Scenario Validation Results 1

Split Ratio	Precision	Recall (TPR)	FPR	F1 Score	ROC AUC	AUCPR
90:10	0.1379	0.1176	0.0013	0.1270	0.9569	0.0828
80:20	0.2286	0.2319	0.0014	0.2302	0.9637	0.1480
70:30	0.2222	0.1942	0.0012	0.2073	0.9554	0.1160
60:40	**0.2727**	**0.2391**	**0.0011**	**0.2548**	**0.9503**	**0.1576**

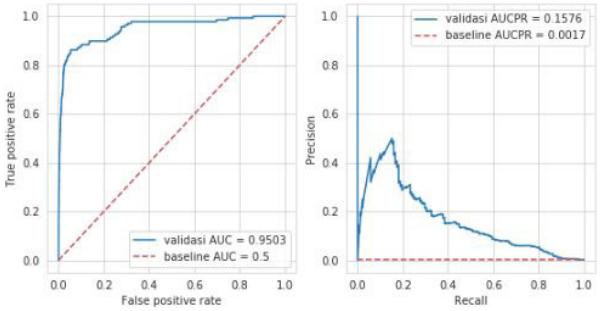

Figure 7. ROC curve and Precision-Recall data Validation scenario 1

Table 3. Best Scenario 1 Validation Confusion Matrix

Split ratio	TN	FP	FN	TP
60:40	79520	88	105	33

Table 4. Scenario Testing Results 1

Split ratio	Precision	Recall (TPR)	FPR	F1 Score	ROC AUC	AUCPR
60:40	0.3382	0.3108	0.0011	0.3239	0.9601	0.2138

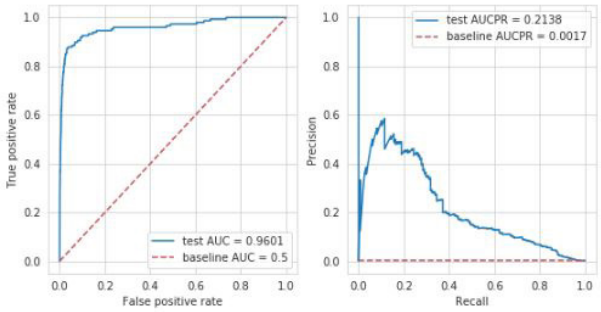

Figure 8. ROC curve and Precision-Recall data Testing scenario 1

Table 5. Best Confusion Matrix Testing Scenario 1

Split ratio	TN	FP	FN	TP
60:40	85295	0	148	0

Table 6. Scenario Validation Results 2

Number of Features	Precision	Recall (TPR)	FPR	F1 Score	ROC AUC	AUCPR
1	0.6160	0.5580	0.0006	0.5856	0.9422	0.5463
2	0.5887	0.5290	0.0006	0.5573	0.9650	0.5061
3	0.6165	0.5942	0.0006	0.6052	0.9684	0.5298
4	0.7231	0.6812	0.0005	0.7015	0.9674	0.6290
5	**0.7254**	**0.7464**	**0.0005**	**0.7357**	**0.9704**	**0.6379**
6	0.6667	0.6522	0.0006	0.6593	0.9731	0.5510
7	0.5500	0.5580	0.0008	0.5540	0.9667	0.4620
8	0.5845	0.6014	0.0007	0.5929	0.9713	0.5277
9	0.4310	0.3623	0.0008	0.3937	0.9714	0.3027
10	0.3798	0.3551	0.0010	0.3670	0.9732	0.2959

To better understand the characteristics between ROCAUC and AUCPR, a confusion matrix for the best model of validation data is presented in Table 3, and a confusion matrix for testing data in Table 4. It appears that a very large TN value compared to TP causes the ROCAUC value tending always to be high. Whereas AUCPR better describes the performance characteristics of the resulting model.

c. Experiment Scenario Results 2

Details of the evaluation results on the validation data with the number of features used can be seen in Table 6. Based on the evaluation results on the validation data, the best model with the number of features used are the 5 best features. The results of evaluating the best model using Testing data can be seen in Table 7. In Figure 9 the best ROC curves and precision Recall models are presented in the validation data and Figure 10 for Testing data.

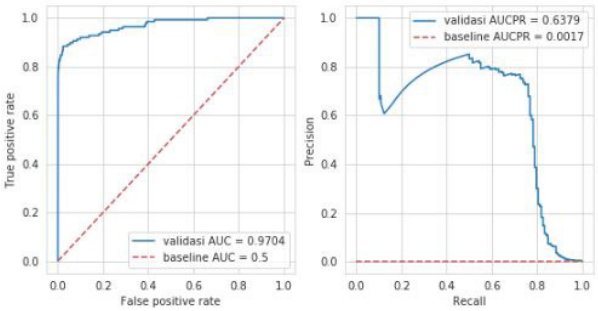

Figure 9. ROC curve and Precision-Recall data Validation scenario 2

Table 7. Scenario Testing Results 2

Number of Features	Precision	Recall (TPR)	FPR	F1 Score	ROC AUC	AUCPR
5	0.7222	0.7905	0.0005	0.7548	0.9732	0.7503

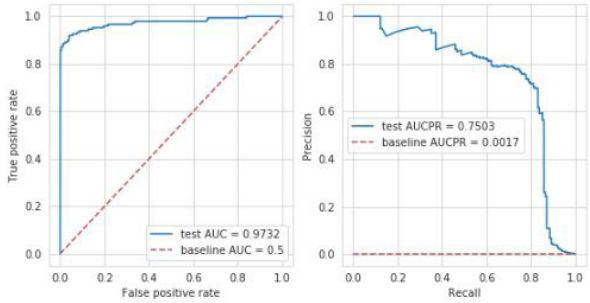

Figure 10. ROC Curve and Precision-Recall Data Testing Scenario 2

Table 8. Scenario Validation Results 3

Fraud Ratio (%)	Precision	Recall (TPR)	FPR	F1 Score	ROC AUC	AUCPR
100	0.7254	0.7464	0.0005	0.7357	0.9704	0.6379
90	0.7368	0.7101	0.0004	0.7232	0.9704	0.6094
80	0.7481	0.7101	0.0004	0.7286	0.9706	0.6294
70	0.7795	0.7174	0.0004	0.7472	0.9704	0.6363
60	**0.8000**	**0.6957**	**0.0003**	**0.7442**	**0.9692**	**0.6429**
50	0.7807	0.6449	0.0003	0.7063	0.9704	0.6049
40	0.7767	0.5797	0.0003	0.6639	0.9706	0.6033
30	0.8026	0.4420	0.0002	0.5701	0.9695	0.6071
20	0.8415	0.5000	0.0002	0.6273	0.9715	0.6463
10	0.7736	0.2971	0.0002	0.4293	0.9719	0.6316
5	0.7209	0.2246	0.0002	0.3425	0.9713	0.6227

d. Experiment Scenario Results 3

Based on the evaluation results on the validation data, the best model with a fraud ratio in the training data is 60%. Details of the evaluation results in the validation data for each variation of the fraud ratio in the training data can be seen in Table 8. The results of the evaluation of the best model using Testing data can be seen in Table 9. Figure 11 shows the best ROC curve and precision Recall model in the validation data and Figure 12 for Testing data.

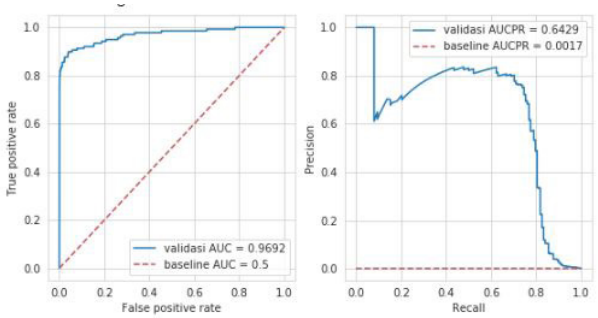

Figure 11. ROC curve and Precision-Recall data Validation scenario 3

Table 9. Scenario Testing Results 3

Fraud Ratio	Precision	Recall (TPR)	FPR	F1 Score	ROC AUC	AUCPR
60	0.7970	0.7162	0.0003	0.7544	0.9751	0.7524

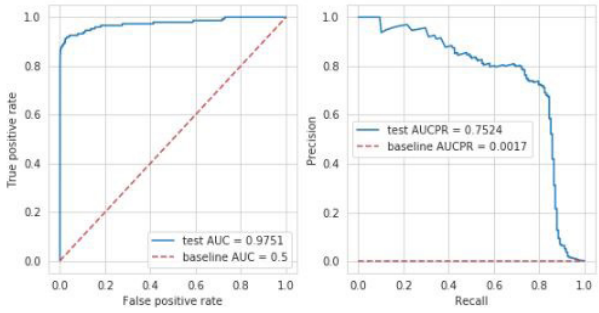

Figure 12. ROC curve and Precision-Recall Data Testing scenario 3

Table 10. Best Model Validation Results for Scenario 4

Precision	Recall (TPR)	FPR	F1 Score	ROC AUC	AUCPR
0.8099	0.7101	0.0003	0.7568	0.9697	0.6380

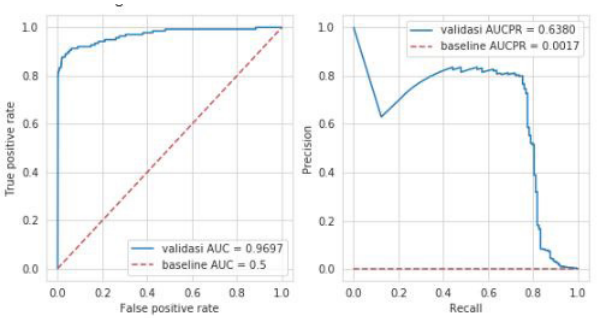

Figure 13. ROC curve and Precision-Recall data Validation scenario 4

Table 11. Scenario Testing Results 4

Precision	Recall (TPR)	FPR	F1 Score	ROC AUC	AUCPR
0.8071	0.7635	0.0003	0.7847	0.9737	0.7592

e. Experiment Scenario Results 4

Through a series of scenario 4 experiments on validation data, it was found that the best hyper-parameter model setting was to use the number of trees 100, maximum sample 128, and contamination 0.001. Evaluation results on the validation data can be seen in Table 10 with Figure 13 for the ROC curve and its precision-recall. Evaluation results on the Testing data for the best model can be seen in Table 11 and Figure 14 presents the ROC curve and precision Recall.

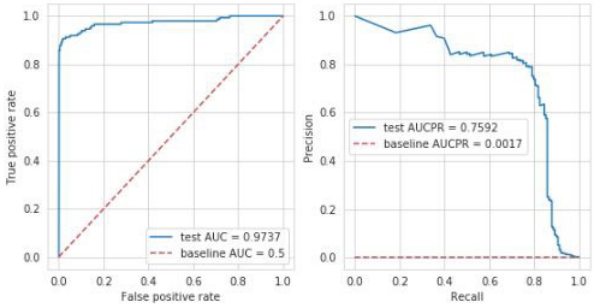

Figure 14. ROC curve and Precision-Recall data Testing scenario 4

4. Discussion

From a series of stages of experiments carried out, it appears that setting the amount of training data affects the quality of the model. This is indicated by obtaining a ratio of 60:40 in the validation data as the best value. Next, the selection of the most important features succeeded in providing a very significant performance improvement from f1-score 0.254826 to f1-score 0.735714. Setting the amount of fraud data in training has also been proven to improve validation performance. The last is setting some hyper-parameters to increase the f1-score validation performance to 0.756757. Overall performance testing data shows good results, even higher than validation data it shows that the resulting model is not overfitting.

In the case of fraud detection in credit card transactions, the value of precision takes precedence over recall considering that companies or organizations generally try to minimize normal customers who have detected fraud (False Positive) because it gives inconvenience so that it can cause loss of the customer. Therefore the final model obtained is prioritized to be able to increase the value of precision with a little compensation in its recall so that the highest precision is obtained by 0.809917, recall 0.710145 with the highest f1-score value is 0.756757.

Table 12. Resume of four experimental scenarios

Scenario	Precision	Recall (TPR)	FPR	F1 Score	ROC AUC	AUCPR
1	0.2727	0.2391	0.0011	0.2548	0.9503	0.1576
2	0.7254	0.7464	0.0005	0.7357	0.9704	0.6379
3	0.8000	0.6957	0.0003	0.7442	0.9692	0.6429
4	0.8099	0.7101	0.0003	**0.7568**	0.9697	0.6380

To compare between ROC and AUCPR, a resume of the four scenarios experiment results can be seen in table 12. It appears that all ROCAUC values are greater than 0.95, although this gives a good impression, it does not reflect the ability of the model to detect actual fraud, i.e., refer to the value precision, recall, and f1-score. In comparison, the AUCPR values from the four scenarios show better compatibility with the values of precision, recall, and f1-score. Thus it is proven that AUCPR is better used as a performance metric for fraud detection on credit card transactions.

5. Conclusion

In this study, we analyze several factors that influence the performance of the Isolation forest model to detect fraud on credit card transactions. There are four experimental scenarios, namely the analysis of the effect of split ratio on data validation, the effect of feature selection, the effect of the amount of fraud data on training data, and the setting of hyper-parameter values. The experimental results show the best results obtained when the data validation split ratio is 60:40, using the five most important features, using 60% of fraud data in training, and setting hyper-parameters with the number of trees 100, maximum sample 128, and contamination 0.001. The best validation results of this model are precision 0.809917, recall 0.710145, f1-score 0.756757, ROC-AUC 0.969728, and AUCPR 0.637993. Based on these models, the results obtained in the testing data are precision 0.807143, recall 0.763514, f1-score 0.784722, ROC-AUC 0.97371, and AUCPR 0.759228. From observations of the results at each stage of the experiment, it can also be concluded that the AUCPR value is better in describing the performance of the model than the ROC-AUC value.

Acknowledgment

The author thanks Diponegoro University, Ministry of Research, Technology, and Higher Education of the Republic of Indonesia for their support of this research.

Reference

[1] S. Gee, *Fraud and Fraud Detection*. Wiley Series, 2015.

[2] ACFE, "Report to the Nation," 2019. [Online]. Available: http://www.acfe.com/rttn.aspx.

[3] Ystats.com, "Global Online Payment Method: Full Year 2016," 2017.

[4] R. Brause, T. Langsdorf, and M. Hepp, "Neural Data Mining for Credit Card Fraud Detection," in *Proceedings of the IEEE International Conference on Tools with Artificial Intelligence*, 1999, pp. 103–106.

[5] A. Dal, O. Caelen, Y. Le Borgne, S. Waterschoot, and G. Bontempi, "Expert Systems with Applications Learned lessons in credit card fraud detection from a practitioner perspective," *Expert Syst. Appl.*, vol. 41, no. 10, pp. 4915–4928, 2014.

[6] S. Bhattacharyya, S. Jha, K. Tharakunnel, and J. C. Westland, "Data mining for credit card fraud : A comparative study," *Decis. Support Syst.*, vol. 50, no. 3, pp. 602–613, 2011.

[7] N. Khare, S. Y. Sait, K. Campus, and K. Campus, "Credit Card Fraud Detection Using Machine Learning Models and Collating Machine Learning Models," *Int. J. Pure Appl. Math.*, vol. 118, no. 20, pp. 825–838, 2018.

[8] F. Braun, O. Caelen, E. N. Smirnov, S. Kelk, and B. Lebichot, "Improving Card Fraud Detection Through Suspicious Pattern Discovery," in *The 30th International Conference on Industrial, Engineering, Other Applications of Applied Intelligent Systems*, 2017, vol. 1, pp. 181–190.

[9] S. Xuan and S. Wang, "Random Forest for Credit Card Fraud Detection," in *2018 IEEE 15th International Conference on Networking, Sensing and Control (ICNSC)*, 2018.

[10] G. E. Melo-Acosta, F. Duitama-Munoz, and J. D. Arias-Londono, "Fraud detection in big data using supervised and semi-supervised learning techniques," *2017 IEEE Colomb. Conf. Commun. Comput. COLCOM 2017 - Proc.*, 2017.

[11] A. Pumsirirat and L. Yan, "Credit Card Fraud Detection using Deep Learning based on Auto-Encoder and Restricted Boltzmann Machine," *Int. J. Adv. Comput. Sci. Appl.*, vol. 9, no. 1, pp. 18–25, 2018.

[12] H. Lee *et al.*, "Feature selection practice for unsupervised learning of credit card fraud detection," *J. Theor. Appl. Inf. Technol.*, vol. 96, no. 2, pp. 408–417, 2018.

[13] V. Zaslavsky and A. Strizhak, "Credit Card Fraud Detection Using Self-Organizing Maps," *Inf. Secur. An Int. J.*, vol. 18, pp. 48–63, 2006.

[14] D. Tripathi, Y. Sharma, T. Lone, and S. Dwivedi, "Credit Card Fraud Detection using Local Outlier Factor," *Int. J. Pure Appl. Math.*, vol. 118, no. 7 Special Issue, pp. 229–234, 2018.

[15] B. Baesens and W. Verbeke, *Fraud Analytics*. Wiley Series, 2015.

[16] N. S. Arunraj, R. Hable, M. Fernandes, K. Leidl, and M. Heigl, "Comparison of Supervised, Semi-supervised and Unsupervised Learning Methods in Network Intrusion Detection System (NIDS) Application," vol. 6, no. 6, 2017.

[17] B. Kristof and S. Rinderle-ma, "Anomaly Detection in Business Process Runtime Behavior – Challenges and Limitations," *CoRR*, vol. abs/1705.0, 2017.

[18] M. Daykin and I. Poole, "A Comparison of Unsupervised Abnormality Detection Methods for Interstitial Lung Disease," in *MIUA2018*, 2018, vol. 3, pp. 1–12.

[19] F. T. Liu and K. M. Ting, "Isolation-Based Anomaly Detection," *ACM Trans. Knowl. Discov. from Data*, vol. 6, no. 1, 2012.

[20] B. Hussain, Q. Du, and P. Ren, "Semi-supervised learning based big data-driven anomaly detection in mobile wireless networks," *China Commun.*, vol. 15, no. 4, pp. 41–57, 2018.

[21] S. Ounacer, H. Ait, E. Bour, Y. Oubrahim, and M. Y. Ghoumari, "Using Isolation Forest in anomaly detection : the case of credit card transactions," vol. 6, no. 2, pp. 394–400, 2018.

[22] X. Niu, L. Wang, and X. Yang, "A Comparison Study of Credit Card Fraud Detection: Supervised versus Unsupervised," *arXiv:1904.10604 [cs.LG]*, 2019.

[23] V. Ceronmani Sharmila, K. R. Kumar, R. Sundaram, D. Samyuktha, and R. Harish, "Credit Card Fraud Detection Using Anomaly Techniques," *Proc. 1st Int. Conf. Innov. Inf. Commun. Technol. ICIICT 2019*, pp. 1–6, 2019.

[24] T. Saito and M. Rehmsmeier, "The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets," *PLoS One*, vol. 10, no. 3, pp. 1–21, 2015.

[25] J. Davis and M. Goadrich, "The relationship between precision-recall and ROC curves," *ACM Int. Conf. Proceeding Ser.*, vol. 148, pp. 233–240, 2006.