Application of Clustering Filters in Order to Exclude Irrelevant Instances of the Process Before Using Reinforcement Learning to Optimize Business Processes in the Bank

Andrey A. Bugaenko*
Data Science Director, Sberbank PJSC, Moscow, Russia
*Corresponding author:
Email: germes86@mail.ru, aabugaenko@sberbank.ru
ORCID 0000-0002-3372-5652

Article History: Received: 10 November 2020; Revised 12 January 2021 Accepted: 27 January 2021; Published online: 5 April 2021

Abstract

The research offers and describes the use of clustering filters in order to exclude preliminarily the instances of the process, which contain errors, and which are not related directly with the business process, and, accordingly, are irrelevant for the analysis. Comparison of 15 types of filters was performed using mapped-out data. It was shown that successful preliminary filtering is possible before the application of reinforcement learning for business process analysis, which reduces the data processing amount.

Key words

Machine learning, process mining, clustering filters, DBSCAN, SVM, robust covariance, isolation forest, local outlier factor, banking.

Introduction

Currently, when applying reinforcement learning in order to analyze business processes in the bank divisions[14]-[15], we quite often see the situation, when the erroneous actions of managers performing the process, which are not related to the business process directly, are logged in the system and drawn in the network graph of the process as additional nodes, thus distorting the actual time of the completion of the process nodes. This not only distorts the vision of the process, but makes the network graph more complicated, and most importantly, slows down data processing materially.

In case of the analysis of business processes in the bank divisions, there exist two main types of the business process instances, which have no business sense, so it is expedient to filter them.

The first type is represented by the cases, when a manager enters the primary system or a subsystem of the primary system by mistake. In this case he/she exits the primary system (subsystem) immediately or performs several actions and further exits the primary system (subsystem). This type of erroneous instances of the process is characterized by(1):

- short time of the process instance
- small number of nodes of the process in the instance
- absence of the fact of successful completion of the process in the instance (for example, for the process of payment acceptance – absence of the fact of effected payment; for the process of the deposit replenishment – absence of the fact of the flow of funds into the deposit, etc.)

The second type is the cases when a manager, upon completion of the process, does not exit the primary system or leaves the primary system for a while in any of the process nodes, performing the actions, which are not related to the process directly. Such a mistake type is characterized by(2):

- abnormally large time of the process instance
- abnormally large time of one of the nodes of the process instance

The instances with the abovementioned problems are usually not filtered when building the business process model, thus accepting the problem with the "inappropriate" lines in the process model and additional load during processing. This occurs due to the fact that time costs for "filtration" are extremely high and fall short of the benefits of elimination of abnormal instances.

Studies [1]-[13], [16]-[17] consider the application of the process mining methodology for business processes analysis, however none of the sources performs preliminary deletion of "irrelevant" process instances in order to "ease" the sampling. Besides, the studies do not focus on the peculiarities of the business processes in bank divisions.

However if there were a low-cost method, this procedure would be very advantageous. The efficient solution to this problem could be the application of the smart filtration methods (clustering filters).

Research Description

Currently, there exist a large variety of clustering methods, which apply efficiently in order to divide the sample into two classes: "rejection" and "non-rejection." The idea was to try to apply such filters to separate the class of "erroneous" instances of the process both in the event of errors when entering the system, and in the event of errors of timely exit.

The tests used 13 typical business processes in the bank offices: acceptance of individuals' (Is') payments, lodging cash into/withdrawal cash from an account, issue, change and closing of bank cards, sale of coins and bullions, effectuation of Is' bank transfers, regular payments towards the repayment of credits, opening/closing/replenishment of deposits, Is' currency exchange, lease of safe deposit boxes, sale of securities and certificates, cash change and change of spoiled banknotes, expert examination of token money, compensations and payments towards depositors of external banks.

It was these business processes in the bank divisions that were chosen for the analysis as the most frequent ones. Sample of instances was formed and time of the process instance was calculated for each of the processes. Further, the instances were mapped out manually using features (1) and (2). The next step was to test 15 types of filters for each process: Robust Covariance, one-class SVM, isolation forest, local outlier factor, DBSCAN, Robust Covariance + one-class SVM, Robust Covariance + isolation forest, Robust Covariance + local outlier factor, Robust Covariance + DBSCAN, one-class SVM + isolation forest, one-class SVM + local outlier factor, one-class SVM + DBSCAN, isolation forest + local outlier factor, isolation forest + DBSCAN, local outlier factor + DBSCAN.

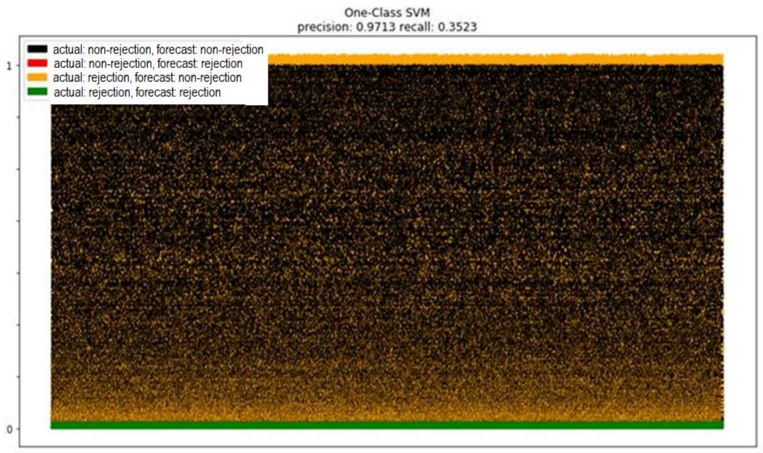
In view of the results, the values of the indices “recall” and “precision” were calculated. Further, the results of these two indices were averaged for all types of business processes as simple arithmetical mean.

A high value of “precision” is critically important for our task; a high value of “recall” is very desirable, but not critical. If we have low “precision”, this means that the filter marks actual instances that are not irrelevant, and we can lose some important case during filtration – some problem in the business process we should find and optimize. It is not important for us to preserve absolutely all problem lines of the process during filtration, as well as to identify them. If “recall” is not high, the algorithm misses the instances to be filtered. Indeed, we still have a senseless line in the network graph of the process, but nevertheless the filter rejects some instances reducing the sample amount and easing the data processing. Actually, “recall” has one restriction only, through which the calculation costs for the operation of the filter itself were “covered” by the number of the filtered irrelevant instances.

Looking ahead, it should be noted that the results showed that for all considered business processes in the bank divisions, the filters showed approximately the same levels of “recall” – “precision”; in other words, the quality of filtration (clustering) depends on the selected method and does not depend on the type of the tested process.

If One-Class SVM (pic.1) is applied, the average value of “recall” for all types of the business processes is 0,3523, “precision” – 0,9713. This means that in the majority of cases the algorithm dos not mark the non-rejection cases as rejection; however, 0,3523 cases of rejection only are identified successfully. Besides, the peculiarity of the algorithm is that it works rather well for the process instances with erroneous enter, but actually does not identify the process instances, where a manager forgot to exit the system, and so it cannot be applied successfully for our task.

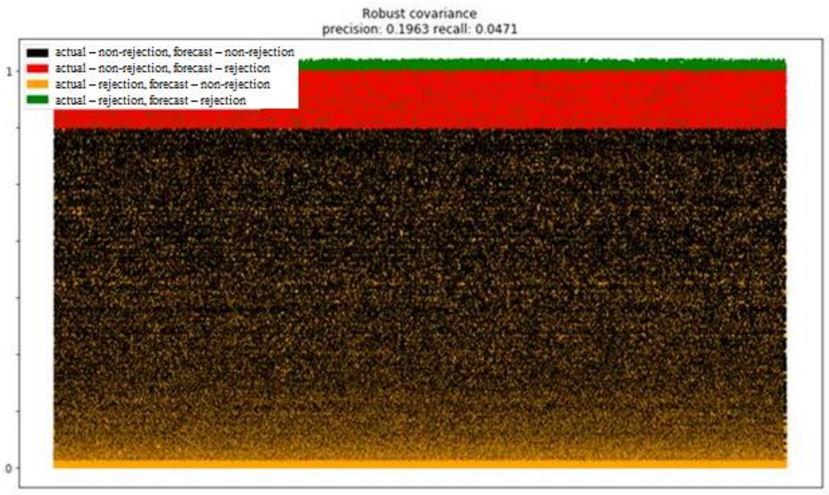
Pic.1. Prediction results by One-Class SVM



The picture shows the normalized distribution of all process instances by all types of the process (the values of the time of performance of each instance of the business process were taken, then they were normalized towards the maximum value of the time of the distribution, further the normalized values of each instance on each type of the process were placed onto one diagram, where Y-axis is the normalized value of the time of the performance, X-axis is a random instance of the sample). The colors show: green: actual – rejection, forecast – rejection; black: actual – non-rejection, forecast – non-rejection; yellow: actual – rejection, forecast – non-rejection; red: actual – non-rejection, forecast – rejection.)

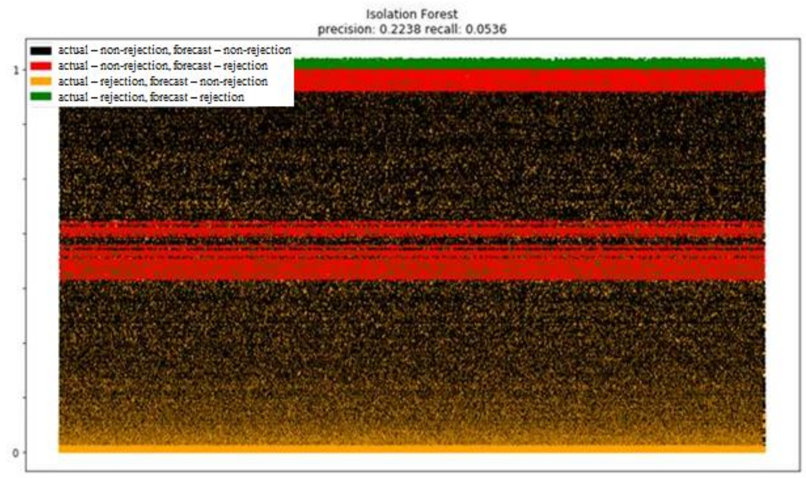
The application of Robust Covariance (pic.2) gives the average value of "recall" for all types of business processes equal to 0,0471, "precision" – 0,1963. In spite of the fact that the algorithm defines successfully some instances of the process, in which a manager forgot to exit the system, it marks by mistake the large number of actual instances, which are not searched for, as abnormality. Besides, the algorithm actually does not mark the manager's erroneous entering the system.

Pic.2. Prediction results by Robust covariance



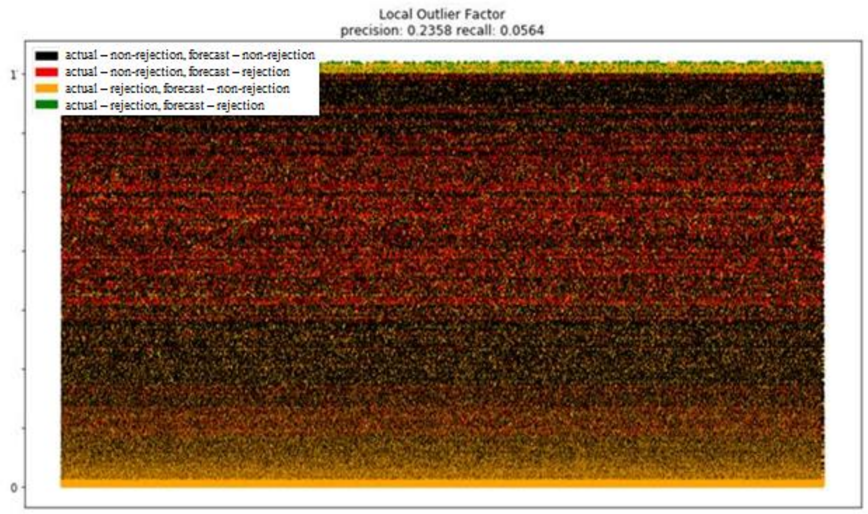
The application of isolation forest (pic.3) shows the same problem, as in robust covariance – the algorithm does not identify the instances of the process with erroneous entering the system. However the average indices of “recall” are 0,0536, average “precision” is 0,2238, which is better than in robust covariance. But nevertheless, the algorithm cannot be used with such values of these indices. After that, besides the large number of false positives under high values, the algorithm marks by mistake the instances with small normalized values of the time of performance as well.

Pic.3. Prediction results by Isolation Forest



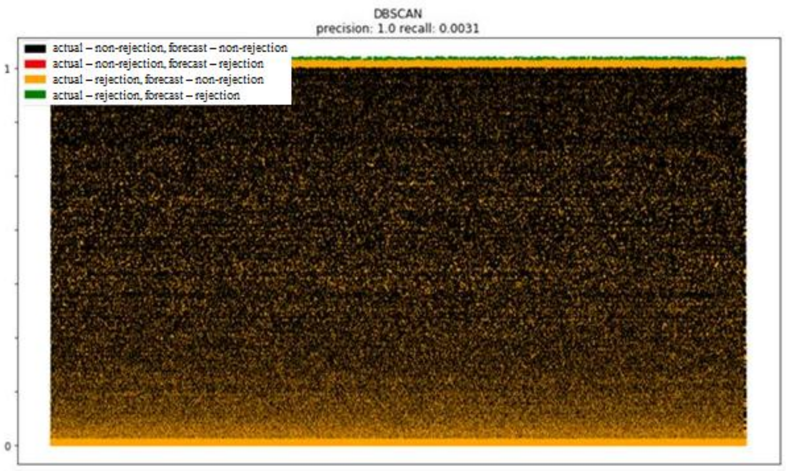
The algorithm Local Outlier factor (pic.4) has the same problem as Robust Covariance and isolation forest – it marks successfully the cases of the manager’s failure to exit the system only. Besides, it identifies their quite insufficient number: “recall” – 0,0564, average “precision” – 0,2358. Further, as opposed to the previous algorithms, the method gives erroneous positives in a large number of cases, and this does not depend on the amount of time of the performance of the process, in contrast to the previous methods.

Pic.4. Prediction results by Local Outlier factor



The application of the method DBSCAN (pic.5), as a result, gives the largest value of “precision” – 1, and the smallest “recall” – 0,0031, out of all the tested algorithms. Since “precision” is the most critical index for us, we could choose DBSCAN for our work, but the method has such a small “recall”, which makes filtration actually irrational for the described types of the cases, because, as a matter of fact, 0,3% only of them will be deleted, and the calculation costs for the application of the filter will be higher, than reduction in the load on the data processing. Besides, the method does not identify the erroneous entering the system.

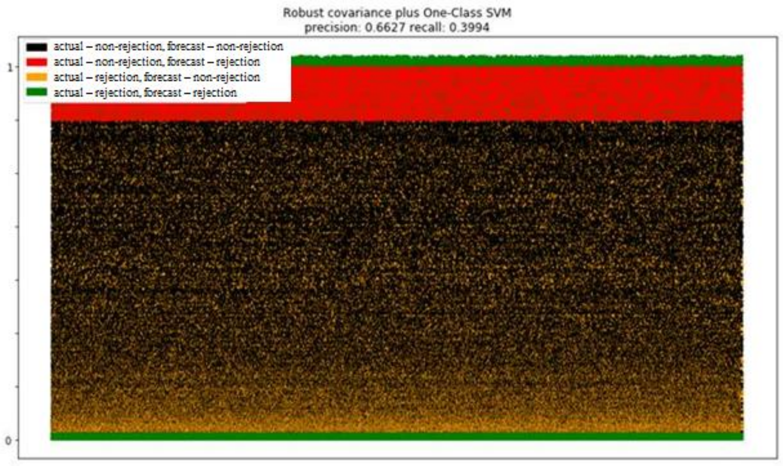
Pic.5. Prediction results by DBSCAN



The analysis showed that none of the checked filtration clustering methods meets our requirements, since one-class SVM identifies one of the two types of the cases only, DBSCAN has extremely low "recall" value, and the remaining three methods have an inappropriately low "precision" value. That is why trying to test mixes of the algorithm combinations makes sense.

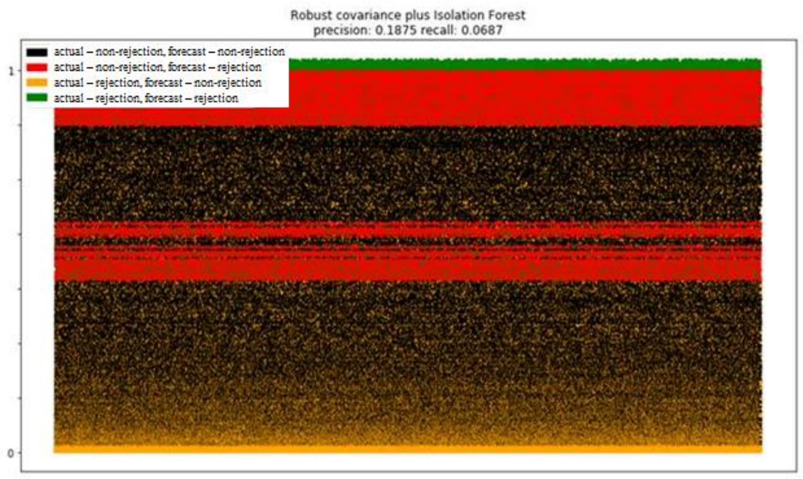
When combining Robust Covariance + one-class SVM (pic.6), the algorithm successfully identifies two types of cases. "Precision" of the method is 0,6627, which is three times better than Robust Covariance's one, but one and a half time worse than one-class SVM's one. "Recall" is 0,3994, which is roughly equivalent to one-class SVM, and by times better than in case of Robust Covariance. The algorithm identifies both types of the cases; however it has an inappropriately low "precision" value.

Pic.6. Prediction results by Robust Covariance + one-class SVM



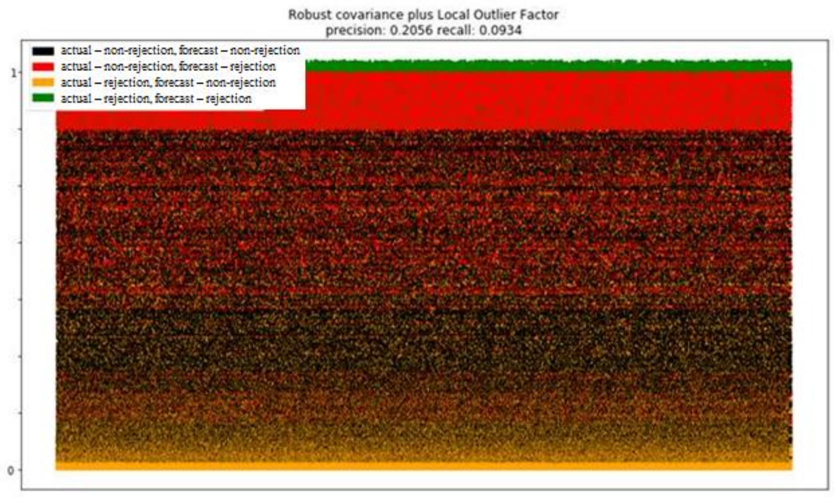
The combination of Robust Covariance + isolation forest (pic.7), like Robust Covariance and isolation forest separately, defines one type of the filtered cases only – the forgotten exits from the system. “Precision” is 0,1875, which is roughly equivalent to Robust Covariance, but lower than in isolation forest. “Recall” is 0,0687, which is similar to the value of the methods used separately. The algorithm is not fit for the application, since it identifies one type of the “erroneous” instance only and has a lot of false positives.

Pic.7. Prediction results by Robust Covariance + isolation forest



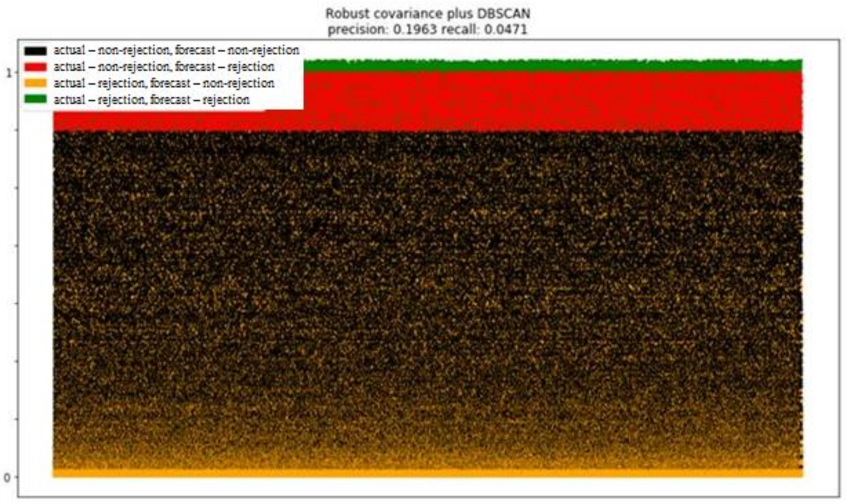
The mix of Robust Covariance + local outlier factor (pic.8), similarly to Robust Covariance and local outlier factor, identifies one type of the filtered cases only. “Precision” is 0,2056, which is roughly equivalent to the methods used separately. “Recall is 0,0934, which is approximately two times better that in the methods taken separately. In a similar way, the algorithm is not fit for application, since it identifies one type of the “erroneous” instance only and has a low level of “precision.”

Pic.8. Prediction results by Robust Covariance + local outlier factor



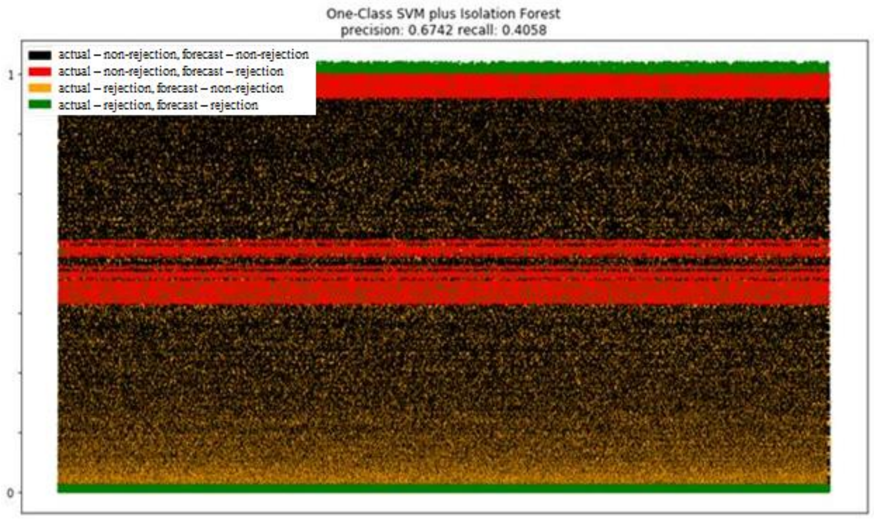
When combining Robust Covariance + DBSCAN (pic.9), the algorithm does not find the cases of managers’ erroneous entering the IT-system. “Precision” of the method is 0,1963, which is similar to Robust Covariance, but five times lower, then in DBSCAN. “Recall” is 0,0471, Robust Covariance has approximately the same value, however this index is ten times lower in DBSCAN. The algorithm is not fit for the application, since it identifies one type of the "erroneous" instance only and has a lot of false positives.

Pic.9. Prediction results by Robust Covariance + DBSCAN



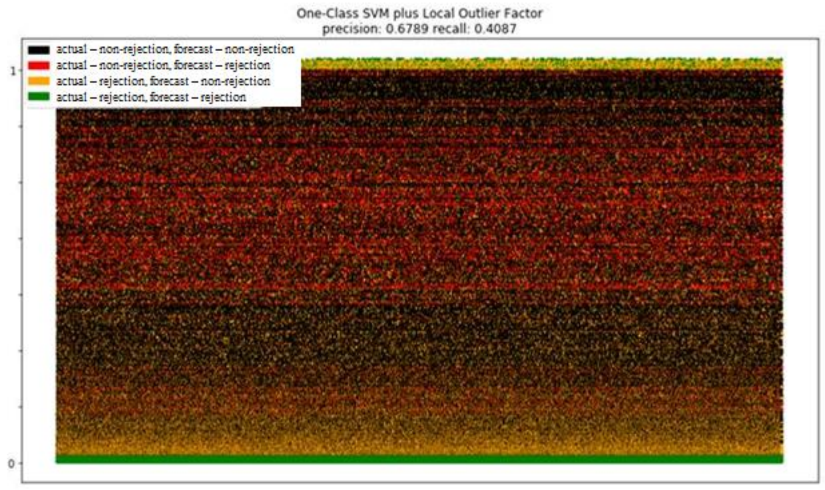
The combination of one-class SVM + isolation forest (pic.10) finds both types of the process instances, which are to be filtered. "Precision" is 0,6742, which is one and a half time lower than for one-class SVM, but three times higher than for isolation forest. "Recall" is 0,4058, which is slightly higher than for one-class SVM and eight times higher than for isolation forest. In spite of the fact that the mix of the methods finds both types of the rejection, it has the level of "precision" which is inadequate for the application.

Pic.10. Prediction results by one-class SVM + isolation forest



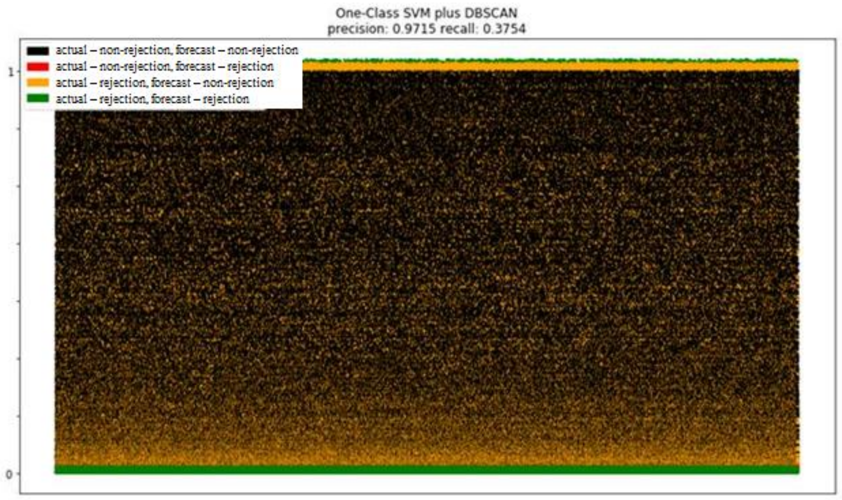
The combination of one-class SVM + local outlier factor (pic.11) is able to find both types of the rejections. "Precision" of the mix is 0,6789, which is three times higher than for local outlier factor, but one and a half time lower than for one-class SVM. It should be noted that the algorithm has the largest index "recall" out of all fifteen studied methods, equal to 0,4087, which is higher than in both methods separately. But nevertheless, in spite of the successful detecting of both types of the rejections and the best index "recall", the mix has an adequately low level of "precision."

Pic.11. Prediction results by one-class SVM + local outlier factor



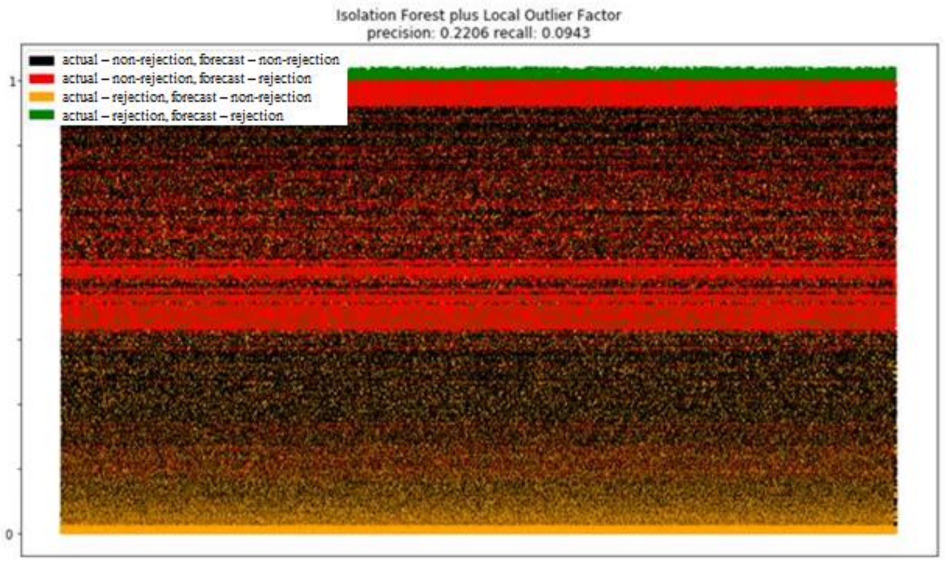
When combining one-class SVM + DBSCAN (pic.12), the algorithm successfully finds the cases of managers' erroneous entering the IT-system, as well as the cases when the managers forget to exit. "Precision" of the mix equals 0,9715, this means that less than 3% of the process instances, which need considering after filtration, will be lost, which is acceptable. "Recall" is 0,3754, which is approximately on the level of one-class SVM, but by times higher than for DBSCAN. The method finds both types of the "rejections", has acceptable values of "precision" and "recall", which makes it absolutely applicable for our task.

Pic.12. Prediction results by one-class SVM + DBSCAN



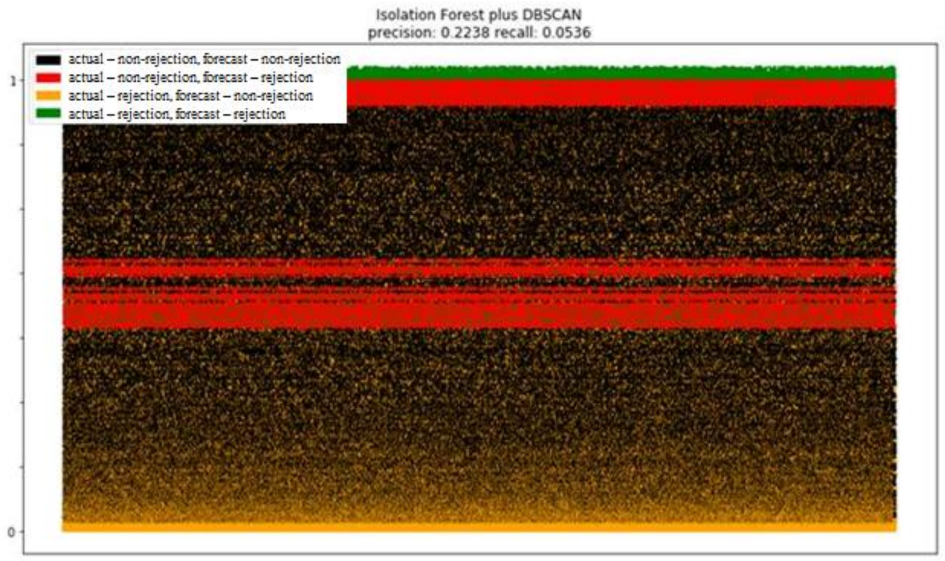
The combination of isolation forest + local outlier factor (pic.13), like both methods separately, finds one type of the "rejection" only. "Precision" of the mix is 0,2206, on the same level as for the methods separately; "recall" is 0,0943, which is approximately twice as large as for the methods separately. The algorithm does not find both types of "rejections", and does not meet needs as to the level of "precision."

Pic.13. Prediction results by isolation forest + local outlier factor



The combination isolation forest + DBSCAN (pic.14) finds one type of the "rejections" only. "Precision" of the mix is 0,2237, which is approximately the same as for isolation forest and five times lower than for DBSCAN; "recall" is 0,0536, which is as well similar to isolation forest, but ten times higher than for DBSCAN. The requirements to the detection and error level are not met.

Pic.14. Prediction results by isolation forest + DBSCAN



The combination local outlier factor + DBSCAN (pic.15) finds one type of the "rejection" only. "Precision" of the mix is 0,2398, "recall" is 0,0577, which has the same value, as in local outlier factor. The algorithm does not find both types of the "rejections", and it does not meet needs as to the level of "precision."

Pic.15. Prediction results by local outlier factor + DBSCAN

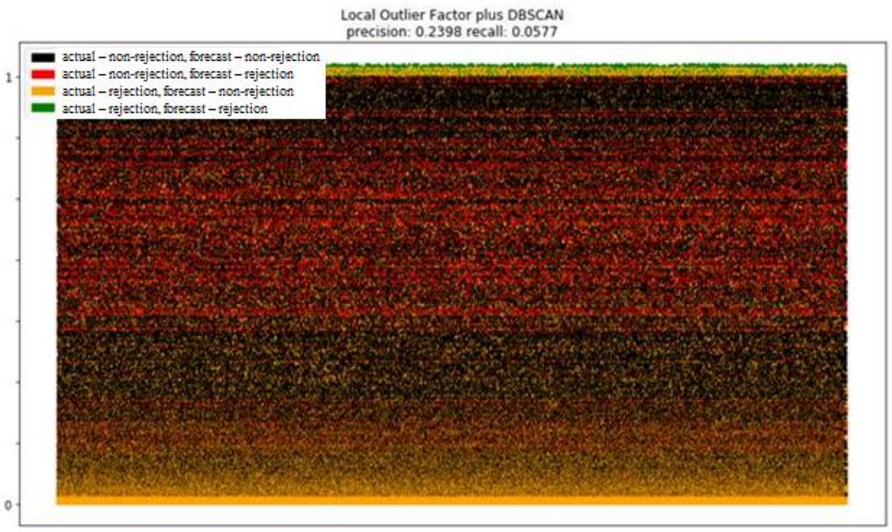


Table.1 Comparison of methods for determining irrelevant cases

	Detection of erroneous entries in the IT system	Detecting forgotten exits from an IT system	Precision	Recall
Robust Covariance,	No	Yes	0,1963	0,0471
one-class SVM,	Yes	No	0,9713	0,3523
isolation forest,	No	Yes	0,2238	0,0563
local outlier factor,	No	Yes	0,2358	0,0564
DBSCAN,	No	Yes	1	0,0031
Robust Covariance + one-class SVM,	Yes	Yes	0,6627	0,3994
Robust Covariance + isolation forest,	No	Yes	0,1875	0,0687
Robust Covariance + local outlier factor,	No	Yes	0,2056	0,0934
Robust Covariance + DBSCAN,	No	Yes	0,1963	0,0471
one-class SVM + isolation forest,	Yes	Yes	0,6742	0,4058
one-class SVM + local outlier factor,	Yes	Yes	0,6789	0,4087
one-class SVM + DBSCAN,	Yes	Yes	0,9715	0,3754
isolation forest + local outlier factor,	No	Yes	0,2206	0,0943
isolation forest + DBSCAN,	No	Yes	0,2238	0,0536
local outlier factor + DBSCAN.	No	Yes	0,2398	0,0577

The final summary of the applicability of this or that method looks as follows and shows that the mix of one-class SVM + DBSCAN is the only method of clustering filtration out of the considered ones that suits our task (table.1).

Conclusion

The study offers the way of reducing the sampling amount when processing the instances of business processes in the process mining methodology. It is offered to reduce the sampling through deletion of two types of irrelevant cases: when a manager enters IT system by mistake and when a manager forgets to exit, so the process instance shows abnormal time. The study has analyzed fifteen methods of clustering filtration in thirteen typical business processes in the bank divisions. The results show the following conclusions:

- For all types of the business processes, clustering filters showed roughly the same results. The obtained characteristics "recall" and "precision" actually do not depend on the type of the business process: they depend on the method.
- Robust Covariance, isolation forest, local outlier factor, DBSCAN, Robust Covariance + isolation forest, Robust Covariance + local outlier factor, Robust Covariance + DBSCAN, isolation forest + local outlier factor, isolation forest + DBSCAN, local outlier factor + DBSCAN do not identify the situation, when a manager enters the system by mistake.
- one-class SVM does not identify the cases, when a manager forgets to exit the system, and we see an abnormally long instance of the process.
- Robust Covariance, isolation forest, local outlier factor, Robust Covariance + one-class SVM, Robust Covariance + isolation forest, Robust Covariance + local outlier factor, Robust Covariance + DBSCAN, one-class SVM + isolation forest, one-class SVM + local outlier factor, isolation forest + local outlier factor, isolation forest + DBSCAN, local outlier factor + DBSCAN showed an unacceptably low level of "precision"; as a consequence, we can filter the process instances, which are not rejections, and miss the moments, which are really important for the business process optimization.
- Robust Covariance, isolation forest, local outlier factor, DBSCAN, Robust Covariance + isolation forest, Robust Covariance + local outlier factor, Robust Covariance + DBSCAN, isolation forest + local outlier factor, isolation forest + DBSCAN, local outlier factor + DBSCAN showed an unacceptably low level of "recall", which makes their application inefficient, since calculation costs for the use of a filter turn out to be higher than calculation costs for processing of additional "irrelevant" instances of the process.
- The mix of the filters one-class SVM + DBSCAN copes with the task excellently, identifying both types of "irrelevant" instances of the process, and having the level of "precision", which is close to one, "filters" more than one-third of such instances, reducing costs for processing.

However, within the framework of the current task, there is a potential for further studies. The application of the process mining methodology often involves a task of defining an optimal sequence of a manager's actions in the business process. The application of reinforcement learning (RL) seems to be very efficient for this task. In particular, definition of the RL method which is the best for this task, description of the methods of creation of the environment, loss and reward, and consideration of the peculiarities of setting hyperparameters are of interest. Another line of this task is search for quick wins. The analysis of the business processes structure obtained from logs requires a lot of time and efforts. The use of the machine learning methods for the automated analysis and identification of “bottlenecks” in the business process seems to be very promising.

References

[1]. Process Mining: Discovery, Conformance and Enhancement of Business Processes by W.M.P. van der Aalst, Springer Verlag, 2011 (ISBN 978-3-642-19344-6).

[2]. Discovering more precise process models from event logs by filtering out chaotic activities, Tax, N., Sidorova, N., van der Aalst, W.M.P, Journal of Intelligent Information Systems 2019

[3]. Wil van der Aalst , Using process mining to deal with “events” rather than “numbers”?, Spreadsheets for business process management, 15 January 2017

[4]. Wil van der Aalst, Process Mining, Data Science in Action, Second Edition, Springer , 2016

[5]. Born, M., Brelage, C., Markovic, I., Pfeiffer, D., Weber, I.: Auto-completion for executable business process models. In: Ardagna, D., Mecella, M., Yang, J. (eds.) BPM 2008. LNBIP, vol. 17, pp. 510–515. Springer, Heidelberg (2009).

[6]. Lin, F.R., Pai, Y.H.: Using multi-agent simulation and learning to design new business processes. IEEE Trans. Syst. Man Cybernet. Part A (Syst. Hum.) 30(3), 380–384 (2000)

[7]. Jianmin Wang, Raymond K. Wong, Jianwei Ding, Qinlong Guo and Lijie Wen “Efficient selection of mining algorithm”, IEEE. Transactions on Services Computing 01/2013.

[8]. Genetic Process Mining , W.M.P. van der Aalst, A.K. Alves de Medeiros, and A.J.M.M. Weijters, Department of Technology Management, Eindhoven University of Technology, P.O. Box 513, NL-5600 MB, Eindhoven, The Netherlands.

[9]. Sarno, R., Sinaga, F., Sungkono, K.R. , Anomaly detection in business processes using process mining and fuzzy association rule learning, 2020 Journal of Big Data

[10]. Aghabaghery, R., Hashemi Golpayegani, A., Esmaeili, L., A new method for organizational process model discovery through the analysis of workflows and data exchange networks, 2020 Social Network Analysis and Mining, 10(1),12

[11]. De Oliveira, H., Augusto, V., Jouaneton, B., Prodel, M., Xie, X., Optimal process mining of timed event logs, 2020 Information Sciences, 528, c. 58-78

[12]. Diba, K., Batoulis, K., Weidlich, M., Weske, M., Extraction, correlation, and abstraction of event data for process mining, 2020 Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 10(3)

[13]. Andrews, R., van Dun, C.G.J., Wynn, M.T., Röglinger, M.K.E., ter Hofstede, A.H.M., Quality-informed semi-automated event log generation for process mining, 2020 Decision Support Systems, 132

[14]. Panfilov M., Goncharenko I., Bugaenko A., Application of DS methods for solving applied task in finance, AI Journey conference, Kaliningrad, 2019

[15]. Bugaenko A., Application of machine learning for post process mining analysis and problem detection in bank, Advances in Intelligent Systems and Computing, April 2021

[16]. Dijkman, R., Gao, J., Syamsiyah, A., Grefen, P., ter Hofstede, A., Enabling efficient process mining on large data sets: realizing an in-database process mining operator, 2020 Distributed and Parallel Databases, 38(1), c. 227-253

[17]. Leemans, S.J.J., Fahland, D., Information-preserving abstractions of event data in process mining, 2020 Knowledge and Information Systems, 62(3), c. 1143-1197