data

MDPI

Article

Extraction of Missing Tendency Using Decision Tree Learning in Business Process Event Log

Hiroki Horita *,†, Yuta Kurihashi † and Nozomi Miyamori †

Graduate School of Science and Engineering, Ibaraki University, Ibaraki 310-8512, Japan; kuragetilyuusa0628@gmail.com (Y.K.); 19nm730f@vc.ibaraki.ac.jp (N.M.)

* Correspondence: hiroki.horita.is@vc.ibaraki.ac.jp; Tel.: +81294-38-5156

† Current address: Hitachi, Ibaraki 316-8511, Japan.

Received: 13 August 2020; Accepted: 7 September 2020; Published: 9 September 2020

Abstract: In recent years, process mining has been attracting attention as an effective method for improving business operations by analyzing event logs that record what is done in business processes. The event log may contain missing data due to technical or human error, and if the data are missing, the analysis results will be inadequate. Traditional methods mainly use prediction completion when there are missing values, but accurate completion is not always possible. In this paper, we propose a method for understanding the tendency of missing values in the event log using decision tree learning without supplementing the missing values. We conducted experiments using data from the incident management system and confirmed the effectiveness of our method.

Keywords: process mining; business process management; data quality; data management

1. Introduction

Information systems encourage the efficient execution and management of business processes. If the results of business process execution are recorded as an event log, we can effectively use it to improve the business process. The analysis of event logs is called process mining [1] and has attracted much attention in recent years. For example, process discovery [1] is a technique to automatically generate a business process model that satisfies the behavior by inputting an event log that records who executed what activity at what time. The visualization of business processes by using business process models can be used to understand the current situation. In addition to this, process mining techniques can also be used to check whether a process conforms to the organization's rules, analyze process performance, and suggest process improvements. The availability of event logs will allow for evidence-based analysis and increase opportunities for business process improvement [2].

Many process mining algorithms assume that the input event log is of high-quality. That is, it is required to have no missing values. However, due to technical (deviations could occur even for automatic logging systems due to machine breakdowns, system bugs, and resource constraints [3]) or human (human error) reasons, the event log may contain missing values [4]. A certain level of errors in event logs is often unavoidable, particularly when event logs are built by integrating several heterogeneous data sources or where manual logging is involved [5]. In addition to this, data failures also occur for the reason that they improve the adaptability of behavior during the execution of a process instance [6].

In data analysis, this phenomenon is called "garbage in, garbage out" [7] and analyzing poor quality data will only yield meaningless results [8,9]. Mans et al. have also shown that the quality of the event log is an important success factor for process mining projects [5]. Therefore, it is necessary to pre-process the event log before analyzing the data.

In the area of process mining, research on pre-processing of event logs has been conducted. Sim et al. proposed a likelihood-based Multiple Imputation to complete the missing values for events in the event log [10]. Conforti et al. proposed a method for repairing the timestamp with which an event was executed in the event log [11]. By using these methods, it is possible to complement the missing value in the event log with the predicted value. However, even if a method of repairing missing values is used, it is not always possible to guarantee that the missing parts are repaired correctly to what is truly executed, which may lead to erroneous repairs. Therefore, it is desirable to prevent missing elements in the data at the time of data acquisition. References [12,13] state that it is important to know how to systematically identify the root causes of data quality problems in event logs. Quality of data can be improved by (i) improving the way in which data are captured while they are being generated and (ii) improving the data after they have been acquired [8]. The above studies [10,11] are (ii), while our study is (i). By using these two perspective methods together, it is expected that data quality can be further improved.

In this paper, we do not repair the missing values, but we propose a method to understand the tendency of the missing events in the event log. By using decision tree learning to learn the information around the missing points in the event log, we can identify the tendency of the missing points from the branching of the constructed tree. Our proposed method is superior in that it is simple and easy for users to understand, considering the purpose of user support for process mining. Furthermore, in the absence of our method, a human would need to look closely at the data to see where the missing values are occurring, and it would be ad hoc. On the other hand, since our method can express the tendency of the occurrence of missing values in the event log by using a decision tree, we think that our method can analyze the cause of the missing values efficiently. We conducted experiments using real data of business processes published by Volvo IT Belgium and confirmed the effectiveness of our method.

This paper is organized as follows. Section 2 explains the preliminary knowledge of this paper. Section 3 explains the proposed method. Section 4 evaluates the proposed method. Section 5 describes the related works. Section 6 summarizes this paper.

2. Background Knowledge

Section 2.1 describes about event logs. Section 2.2 describes about data quality in process mining.

2.1. Event Logs

Event Log is recorded by information systems as a result of events executed in business processes. Performed events are recorded in a business process instance (from the start to the end of a case). A trace is a division of executed events into business process instances, and a trace T_i of length n is represented as an ordered set of events e in the following manner. i represents the identifier of the trace in the event log.

$$T_i = \langle e_{i1}, e_{i2}, \ldots, e_{in} \rangle$$

This indicates that the events were executed in sequence from left to right. Each event can also have attributes such as the person who executed the event and the time it was executed, but we do not use these information in this paper. Each event can be represented by an activity (for example, a register request in the handling of requests for compensation) that represents the content of each event in an easy-to-understand manner. Table 1 is an example of an event log. Trace 1 consists of four events, <e_{11}, e_{12}, e_{13}, e_{14}>, and each event has an activity label <A, B, C, D>. In addition, the person who executed the event (the resource) and the time of execution (timestamp) are recorded. In this paper, we extract missing tendencies with a particular focus on activity.

Table 1. A fragment of an event log.

Trace ID	Timestamp	Event	Activity	Resource	···
1	2019/10/5 11:31	e_{11}	A	Bob	···
1	2019/10/6 12:01	e_{12}	B	Alice	···
1	2019/10/6 15:05	e_{13}	C	Alice	···
1	2019/10/6 17:05	e_{14}	D	Bob	···
2	2019/10/8 10:03	e_{21}	A	Bob	···
2	2019/10/9 11:03	e_{22}	B	Alice	···
2	2019/10/9 15:03	-	-	Bob	···
2	2019/10/10 11:03	e_{24}	D	Alice	···
⋮	···	···	···	···	···

2.2. Data Quality

Ensuring data quality is a key challenge for successful process mining [1]. Reference [14] defines five levels of maturity for event log data quality from excellent-quality (⋆⋆⋆⋆⋆) to low-quality (⋆). Process mining techniques are considered to be applicable to ⋆⋆⋆⋆⋆, ⋆⋆⋆⋆, and ⋆⋆⋆ , and in principle, process mining techniques can be applied to ⋆⋆ and ⋆ event logs as well. However, analysis using such a low-quality event log is problematic and the results cannot be trusted. Therefore, this study proposes a method to find the problems of the current low-quality event logs in order to obtain high-quality event logs in the future at the stage of acquiring event logs in the situation where only ⋆⋆ or ⋆ level event logs are available.

There are various dimensions of data quality [15]. In particular, the reference [4] lists the following four dimensions related to data quality in process mining.

(1) Missing Data: This corresponds to the situation where data that should be recorded is missing. For example, events and attributes in the log and their relationships are lost. Missing values are caused by problems with the logging system or human error.

(2) Incorrect Data: This corresponds to the situation where the log is different from what should be recorded. For example, the entity or value recorded in the log are incorrect.

(3) Imprecise Data: This corresponds to the case where the recorded data is coarse. If such data were recorded, we would not be able to perform the analysis that requires more accurate data. For example, if the timestamp unit is days, it is difficult to analyze in hours or minutes.

(4) Irrelevant Data: This is when more data are recorded that are not of current interest than are needed. Filtering irrelevant data is difficult and is an important issue in process mining.

This paper deals with the above (1) Missing Data, especially for situations where events and activities are missing. Missing events and activities are events and activities that should be recorded but are not. For example, a trace that should be recorded as <A, B, D, E> is recorded as <A, B, -, E>. A hyphenation (-) indicates that an event and activity is missing. When activity D is missing and becomes <A, B, E>, i.e., it may not be detected as missing, this paper focuses on the case where it can be detected as <A, B, -, E>, as in Reference [16].

3. Proposed Method

In this section, we describe a method for extracting the common tendency of traces that contain missing values in an activity using decision tree learning. Figure 1 is an overview of the proposed method. In general, the results of business process execution are recorded in an event log either by the information system or manually. Since the event log may contain missing values due to technical or human causes, we use a decision tree to extract the missing tendency. The proposed method consists of two phases: (1) vectorization of traces containing missing values of activities, and (2) construction and interpretation of decision trees.

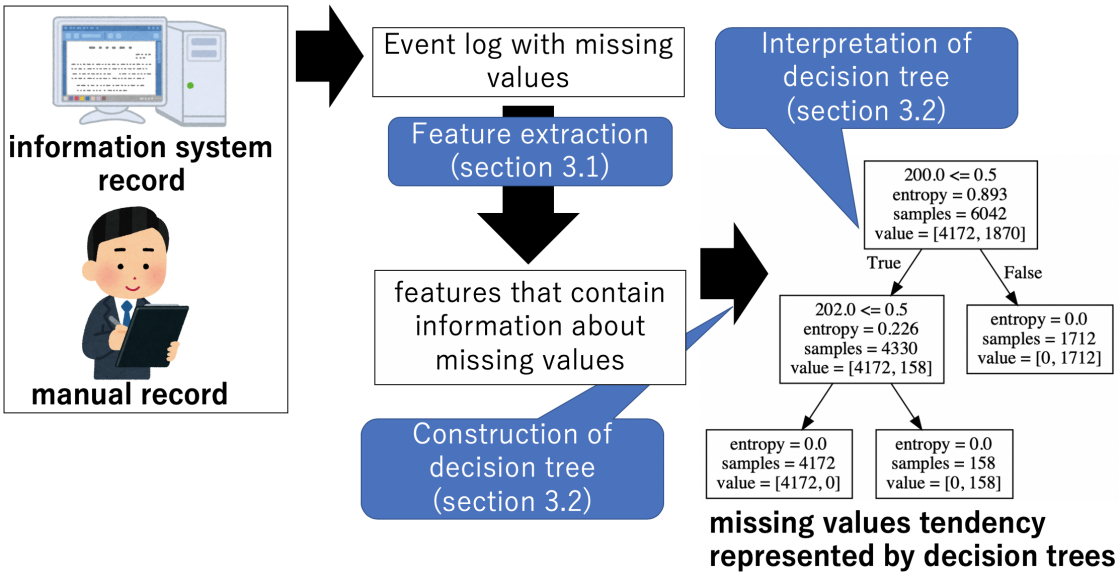

Figure 1. Overview of the proposed method.

3.1. Vectorization of Traces with Missing Activity

The usual event log is in the form of XES (eXtensible Event Stream), an XML-based format, and it is difficult to train a decision tree in this format. In existing research using machine learning in process mining, several methods have been proposed to extract features from the event log for training [17]. However, no method for training the event log including the missing values has been established. In this paper, we focus on the activities performed before and after the missing values in the trace and convert them to a form that can be used for decision tree learning by vectorizing the trace containing the missing values.

Algorithm 1 is an algorithm for generating a set of features from an event log that contains missing values. All the activities in all the traces (from the trace of ID 0 to the trace of ID n) are checked in sequence, and if a missing value is found, the ID of the activity executed immediately before the missing value is added to 200 and the ID of the activity executed immediately after the missing value is added to 100 to get the feature value. Here, we assume that the original types of activities are less than 100. The numbers 100 and 200 are added to the activity IDs in order to distinguish them from the original activities. It does not matter if the numbers to be added are different, such as 1000 and 2000. We also assume that the number of activities is less than or equal to 100 because if there are more than 100 activities, we cannot distinguish, for example, between features added to ID1 plus 100 and the originally existing ID101. If there are more than 100 activities, it is possible to handle them by increasing the number added to the ID. For example, if the number to be added is 200, our method can accommodate up to 200 activities. After generating a set of features, each trace is checked and a one-hot vector is generated, which is set to 1 if the corresponding feature exists and 0 if not.

Table 2 shows an example of generating a vector from traces. In TraceID T2, 4 and 3 are executed immediately before and after the missing value represented by "-". Therefore, adding 200 to 4 adds 204 features, and adding 100 to 3 adds 103 features. The values of features 103 and 204 in TraceID T2 are 1. On the other hand, TraceID T1 is zero because there is no place corresponding to features 103 and 204. The label is set to 1 if each trace contains a missing value and 0 otherwise.

Algorithm 1 Generation of feature set focusing on missing values

Input:
$trace_0, ..., trace_n$: n traces represented as arrays
$trace_i = \langle activity_{i0}, ..., activity_{im} \rangle$: A trace is made up of a collection of activities arranged in order
$FeatureSet = \phi$: set of features
for $i = 0$ to n **do**
  for $j = 0$ to m **do**
    if $(activity_{ij} = \phi)$ **then**
      if $((activity_{ij-1} + 200) \notin FeatureSet)$ **then**
        $FeatureSet$ += $(activity_{ij-1} + 200)$
      end if
      if $((activity_{ij-1} + 100) \notin FeatureSet)$ **then**
        $FeatureSet$ += $(activity_{ij+1} + 100)$
      end if
    end if
  end for
end for
return $FeatureSet$

Table 2. Feature vectors focusing on missing values.

		Features of the Missing Value Relationship			
TraceID	**Trace**	**103**	**204**	···	**Label**
T1	<0,0,1,2,3>	0	0	···	0
T2	<0,1,4,-,3>	1	1	···	1
⋮	⋮	⋮	⋮	⋮	⋮

3.2. Construction and Interpretation of Decision Trees

The event log vectored by the method described in Section 3.1 is turned into a decision tree using the decision tree learning algorithm, CART [18]. Decision tree learning is a method for generating a tree classifier given a certain objective and explanatory variable and dividing the data into a top-down list. In this paper, we learn a decision tree with the objective variable as a variable that indicates whether the trace contains missing values or not, and the explanatory variable as information about the activity taking place before and after the missing values.

Figure 2 shows an example of a decision tree constructed by the proposed method. The "200.0 <= 0.5" in the root node represents the branching condition of the trace, and when the feature 200.0 is less than or equal to 0.5, we assume a one-hot vector with a feature value of 0 or 1, so when the value of the feature 200.0 is 0, the trace branches to the left and If the value of the feature 200.0 is 1, the trace branches to the right. The reason why we chose 0.5 is because of the specification of scikit-learn, the machine learning library we used. It does not matter if the number is not 0.5, as we need to know that the feature of interest is 0 or 1. Samples represent the number of traces, where the number to the left of value represents the number of traces with no missing values and the number to the right of value represents the number of traces with missing values. In this example, 554 traces do not contain any missing values and 387 traces contain them. Entropy is an information-theoretic measure of uncertainty in a multi-set of elements. If the multi-set contains many different elements and each element is unique, then variation is maximal and it takes many "bits" to encode the individual elements [1]. From this decision tree, we can derive the rule of missing values that "there is always

an activity with an activity ID of 0 immediately before the activity with missing values in the trace containing the missing values".

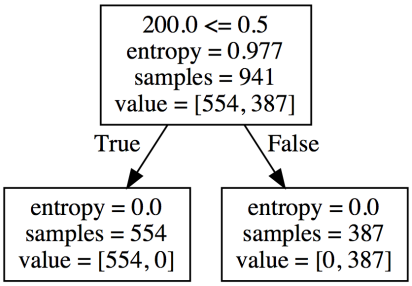

Figure 2. Example of a decision tree constructed by the proposed method.

4. Evaluation

In this section, we describe the experiments to confirm the effectiveness of the proposed method and the results of the experiments.

4.1. Details of the Experiment

As an evaluation experiment of the proposed method, we apply the proposed method to an event log containing missing values, and check whether the tendency of missing values can be extracted by decision tree learning. The reason for this method of evaluation is that there are probably no studies that provide a quantitative comparison with this study.

The data set used was the data from the incident management system published by Volvo IT Belgium, which was the subject of the BPI challenge 2013[1]. The expected workflow of this system represents the process of recovering from the state of the incident to normal service when the incident occurs. The number of traces in this event log is 6042 and there are 13 different types of activities. Table 3 shows the list of activities. Each activity is assigned an ID. From now on, each activity is called by its ID. Because the above event log does not contain any missing values, the following three patterns of missing values were generated separately. The three patterns were determined by focusing on control flow, which is an important factor to consider in business processes. It was randomly decided which IDs were deleted.

Table 3. Activity name and ID.

Activity ID	Activity Name
0	Accepted in-progress
1	Queued-Awaiting-assignment
2	Completed-resolve
3	Accepted assigned
4	Completed closed
5	Accepted wait-user
6	Accepted wait-implementation
7	Accepted wait
8	Completed In Call
9	Accepted wait-vender
10	Accepted wait-customer
11	Unmatched Unmatched
12	Completed cancelled

[1] https://www.win.tue.nl/bpi/doku.php?id=2013:challenge.

pattern 1: The activity immediately before or after the activity with an ID of 0 is deleted and the missing values are generated.

pattern 2: When an activity with an activity ID of 0 or 2 occurs during a trace, the next activity is deleted and a missing value is generated.

pattern 3: When an activity with IDs 5, 6, 7, 9, and 10 is performed during a trace, it generates a random number from 0 to 100. If the generated random number is greater than 40, the activity of IDs 5, 6, 7, 9 and 10 are deleted and the missing value is generated.

After generating the missing values using the method described in Section 4.1, the event log was vectorized and the decision tree was constructed using CART [18], a decision tree learning algorithm of the machine learning library scikit-learn [19].

Our experiments were conducted on an MacBook Air, 1.8 GHz Intel Core i5, each core being equipped with 8 GB main memory, running on macOS 10.13.6.

4.2. Results

We constructed three decision trees corresponding to each of patterns 1, 2, and 3 in Section 4.1 using the proposed method. The results are explained below.

4.2.1. Result of Pattern 1

In pattern 1, the decision tree obtained by applying the proposed method is shown in Figure 3. The value of the root node shows that the number of traces containing the missing values is 3692 in the event log. There are 2350 traces that do not contain missing values. A common rule for traces with missing values is that most of the traces (3589) with missing values have an activity with an activity ID of 0 just before the missing value. Then, it can be seen from the classification rules of the decision tree that in other traces that contain missing values, there is an activity with an activity ID of 0 that is performed immediately after the missing value. From these two rules in the resulting decision tree, we were able to extract a rule for the missing values contained in this event log, which states that an activity with an activity ID of 0 is always performed immediately before or after the missing value. This is equivalent to the missing rules of artificially generated pattern 1, so we can see that we have correctly extracted the rules common to each of the traces containing the missing values.

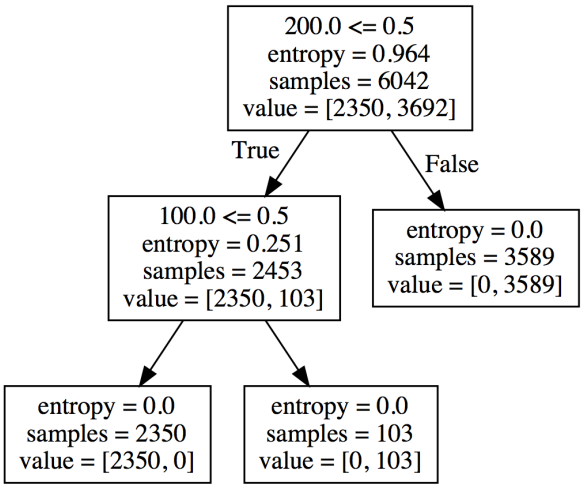

Figure 3. Decision tree in pattern 1.

4.2.2. Result of Pattern 2

In pattern 2, the decision tree obtained by applying the proposed method is shown in Figure 4. The value of the root node shows that the number of traces containing the missing values is 1870 in the event log. There are 4172 traces that do not contain missing values. It can be seen that the rule "an

activity with an activity ID of 0 or 2 is executed just before the missing value" can be extracted using the decision tree shown in Figure 4. This is equivalent to the missing rules of artificially generated pattern 2, so we can see that we have correctly extracted the rules common to the traces containing the missing values.

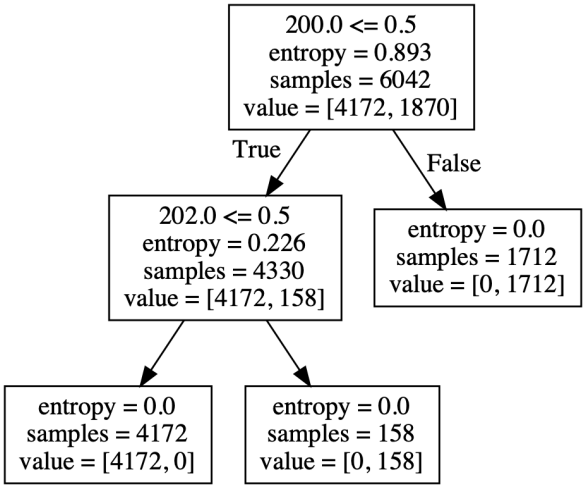

Figure 4. Decision tree in pattern 2.

4.2.3. Result of Pattern 3

In pattern 3, the decision tree obtained by applying the proposed method is shown in Figure 5. The value of the root node shows that the number of traces containing the missing values is 2813 in the event log. There are 3229 traces that do not contain missing values. A common rule for traces containing missing values is that an activity with an activity ID of 0 takes place before the missing values. Furthermore, the fact that an activity with an activity ID of 2 is performed after the missing value is extracted as a rule. This result shows that the activity with an activity ID of 0 was performed before the missing value, and the activity with an activity ID of 2 was often performed after the missing value.

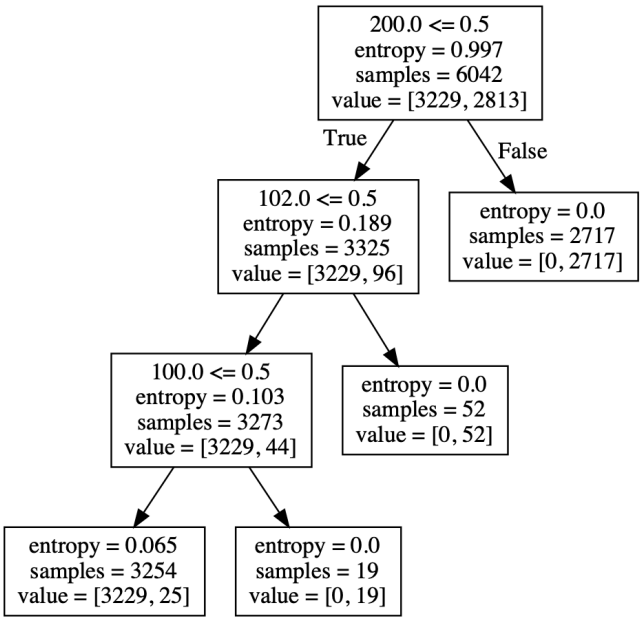

Figure 5. Decision tree in pattern 3.

4.2.4. Summary and Discussion of the Results

In summary, in the case of patterns 1 and 2, where the activity was missing by a definitive rule, we were able to classify with 100% accuracy, but in the case of (3), where the activity was probabilistically missing, we were unable to classify 25 traces correctly. In the case of pattern 3, 25 out of 6042 traces could not be classified correctly, which means that the misclassification of 25 traces occurred.

If our methods are not used, analysts need to look closely at the data to see where the missing values are occurring without getting any assistance. On the other hand, since our method can express the tendency of the occurrence of missing values in the event log by using decision trees, we believe that our method can analyze the cause of the missing values efficiently when the pattern of the missing values is observed as shown in 1 and 2. In the case of pattern 3, a misclassification has occurred. To deal with this problem, we can modify the features and extend the interpretation of the constructed decision tree. This is a subject for future work.

5. Related Works

Various studies have been done on the quality of event logs. Many of these studies have proposed algorithms for repairing defects in the event log, such as the presence of missing values in activities and timestamps [3,10,11,16,20–31]. By using these methods, we can obtain higher quality data that are close to the actual business process from the data with missing values. However, the completion of missing values is not always accurate. There are cases in which the completion of a missing value is different from the actual business process.

While existing methods of missing value completion improve data quality immediately, this study proposed a method to understand the tendency of missing values by focusing on the activity of the business process to support long-term accurate data recording.

The importance of obtaining high quality data is not limited to the field of process mining. Various research papers on data repair (e.g., [32–37]) and survey papers (e.g., [38,39]) exist. These are not concerned with the data quality of the event log in process mining. Since there are various formats of data, it is necessary to have a method that suits the analysis target. In our research, we focus on control flow, which is important in business process, to extract the deficit tendency of activity.

There are also studies that have detected problems related to the quality of event logs [40,41] and proposed evaluation methods [42,43]. These are similar to our study, but our study is different in target because it specializes in tendency extraction of missing value.

Andrews et al. proposed a quality-aware, semi-automated approach to extracting event logs from relational data [44]. Fani Sani et al. showed that many process mining algorithms are difficult to handle really large and noisy event data, and proposed a method for selecting traces by some pre-processing functions that reduce the complexity of the event log [45]. Whereas these studies are concerned with pre-processing to generate high-quality event logs, this study is not about pre-processing but is intended to provide continuous support for obtaining high-quality event logs.

Several domain-specific data quality frameworks [46–48] have been proposed. On the other hand, our method is considered to be applicable to a wide range of domains without domain knowledge.

6. Conclusions

In this paper, we proposed a method for identifying the tendency of missing values of activities in an event log with low-quality of business processes by using decision tree learning, which uses information on activities executed before and after the missing values as features. As a result of evaluation experiments, we showed that it is possible to extract rules for classifying traces with and without missing values with high accuracy.

The future work is to improve the ability to express the extracted rules. In this paper, we focused on the activities performed before and after the missing values and learned them as features. By increasing

the number of features such as the person who performed the activity and the execution time, we are expected to be able to extract the tendency for various missing values.

Author Contributions: Conceptualization and Methodology, Y.K., N.M. & H.H.; Development, Y.K.; Validation, Y.K.; Writing—Original Draft, Yuta Kurihashi & H.H.; Writing—Review & Editing, Y.K., N.M. & H.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Acknowledgments: We would like to thank the members of our laboratory for discussing this study.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Van der Aalst, W.M.P. *Process Mining—Data Science in Action*, 2nd ed.; Springer: Berlin/Heidelberg, Germany, 2016. [CrossRef]
2. van Eck, M.L.; Lu, X.; Leemans, S.J.; van der Aalst, W.M. PM^2: A Process Mining Project Methodology. In Proceedings of the International Conference on Advanced Information Systems Engineering, Stockholm, Sweden, 8–12 June 2015; Springer: Berlin/Heidelberg, Germany, 2015; pp. 297–313.
3. Wang, J.; Song, S.; Zhu, X.; Lin, X. Efficient recovery of missing events. *Proc. VLDB Endow.* **2013**, *6*, 841–852. [CrossRef]
4. Bose, R.J.C.; Mans, R.S.; van der Aalst, W.M. Wanna improve process mining results? In Proceedings of the 2013 IEEE Symposium on Computational Intelligence and Data Mining (CIDM), Singapore, 16–19 April 2013; pp. 127–134.
5. Mans, R.S.; van der Aalst, W.M.; Vanwersch, R.J.; Moleman, A.J. Process mining in healthcare: Data challenges when answering frequently posed questions. In *Process Support and Knowledge Representation in Health Care*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 140–153.
6. Song, W.; Jacobsen, H.A. Static and dynamic process change. *IEEE Trans. Serv. Comput.* **2016**, *11*, 215–231. [CrossRef]
7. Wynn, M.T.; Sadiq, S. Responsible process mining-A data quality perspective. In Proceedings of the International Conference on Business Process Management, Vienna, Austria, 1–6 September 2019; Springer: Cham, Switzerland, 2019; pp. 10–15.
8. Batini, C.; Cappiello, C.; Francalanci, C.; Maurino, A. Methodologies for data quality assessment and improvement. *ACM Comput. Surv. (CSUR)* **2009**, *41*, 1–52. [CrossRef]
9. Cai, L.; Zhu, Y. The challenges of data quality and data quality assessment in the big data era. *Data Sci. J.* **2015**, *14*. [CrossRef]
10. Sim, S.; Bae, H.; Choi, Y. Likelihood-based Multiple Imputation by Event Chain Methodology for Repair of Imperfect Event Logs with Missing Data. In Proceedings of the 2019 International Conference on Process Mining (ICPM), Aachen, Germany, 24–26 June 2019; pp. 9–16.
11. Conforti, R.; La Rosa, M.; ter Hofstede, A. *Timestamp Repair for Business Process Event Logs*; University of Melbourne: Melbourne, Australia, 2018.
12. Emamjome, F.; Andrews, R.; ter Hofstede, A.H. A case study lens on process mining in practice. In Proceedings of the OTM Confederated International Conferences "On the Move to Meaningful Internet Systems", Rhodes, Greece, 21–25 October 2019; Springer: Cham, Switzerland, 2019; pp. 127–145.
13. Emamjome, F.; Andrews, R.; ter Hofstede, A.H.; Reijers, H.A. Alohomora: Unlocking Data Quality Causes Through Event Log Context. In Proceedings of the 28th European Conference on Information Systems (ECIS), Marrakech, Morocco, 15–17 June 2020.
14. Van Der Aalst, W.; Adriansyah, A.; De Medeiros, A.K.A.; Arcieri, F.; Baier, T.; Blickle, T.; Bose, J.C.; Van Den Brand, P.; Brandtjen, R.; Buijs, J.; et al. Process mining manifesto. In Proceedings of the International Conference on Business Process Management, Clermont-Ferrand, France, 29 August–2 September 2011; Springer: Berlin/Heidelberg, Germany, 2011; pp. 169–194.
15. Batini, C.; Scannapieco, M. *Data and Information Quality*; Springer International Publishing: Cham, Switzerland, 2016; p. 43.
16. Xu, J.; Liu, J. A Profile Clustering Based Event Logs Repairing Approach for Process Mining. *IEEE Access* **2019**, *7*, 17872–17881. [CrossRef]

17. Teinemaa, I.; Dumas, M.; Rosa, M.L.; Maggi, F.M. Outcome-oriented predictive process monitoring: Review and benchmark. *ACM Trans. Knowl. Discov. Data* **2019**, *13*, 1–57. [CrossRef]
18. Loh, W.Y. Classification and regression trees. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* **2011**, *1*, 14–23. [CrossRef]
19. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
20. Song, W.; Jacobsen, H.A.; Zhang, P. Self-Healing Event Logs. *IEEE Trans. Knowl. Data Eng.* **2019**. [CrossRef]
21. Nguyen, H.T.C.; Lee, S.; Kim, J.; Ko, J.; Comuzzi, M. Autoencoders for improving quality of process event logs. *Expert Syst. Appl.* **2019**, *131*, 132–147. [CrossRef]
22. Sani, M.F.; van Zelst, S.J.; van der Aalst, W.M. Repairing Outlier Behaviour in Event Logs using Contextual Behaviour. *Enterp. Model. Inf. Syst. Archit.* **2019**, *14*, 1–24.
23. Cheng, H.J.; Kumar, A. Process mining on noisy logs—Can log sanitization help to improve performance? *Decis. Support Syst.* **2015**, *79*, 138–149. [CrossRef]
24. Kong, L.; Li, C.; Ge, J.; Li, Z.; Zhang, F.; Luo, B. An Efficient Heuristic Method for Repairing Event Logs Independent of Process Models. In Proceedings of the 4th International Conference on Internet of Things, Big Data and Security, Heraklion, Greece, 2–4 May 2019.
25. Dixit, P.M.; Suriadi, S.; Andrews, R.; Wynn, M.T.; ter Hofstede, A.H.; Buijs, J.C.; van der Aalst, W.M. Detection and interactive repair of event ordering imperfection in process logs. In Proceedings of the International Conference on Advanced Information Systems Engineering, Tallinn, Estonia, 11–15 June 2018; Springer: Cham, Switzerland, 2018; pp. 274–290.
26. Suriadi, S.; Andrews, R.; ter Hofstede, A.H.; Wynn, M.T. Event log imperfection patterns for process mining: Towards a systematic approach to cleaning event logs. *Inf. Syst.* **2017**, *64*, 132–150. [CrossRef]
27. Song, W.; Xia, X.; Jacobsen, H.A.; Zhang, P.; Hu, H. Heuristic recovery of missing events in process logs. In Proceedings of the 2015 IEEE International Conference on Web Services, New York, NY, USA, 27 June–2 July 2015; pp. 105–112.
28. Rogge-Solti, A.; Mans, R.S.; van der Aalst, W.M.; Weske, M. Improving documentation by repairing event logs. In Proceedings of the IFIP Working Conference on The Practice of Enterprise Modeling, Riga, Latvia, 6–7 November 2013; Springer: Berlin/Heidelberg, Germany, 2013; pp. 129–144.
29. Song, S.; Cao, Y.; Wang, J. Cleaning timestamps with temporal constraints. *Proc. VLDB Endow.* **2016**, *9*, 708–719. [CrossRef]
30. Wang, J.; Song, S.; Lin, X.; Zhu, X.; Pei, J. Cleaning structured event logs: A graph repair approach. In Proceedings of the 2015 IEEE 31st International Conference on Data Engineering, Seoul, Korea, 13–17 April 2015; pp. 30–41.
31. Ly, L.T.; Indiono, C.; Mangler, J.; Rinderle-Ma, S. Data transformation and semantic log purging for process mining. In Proceedings of the International Conference on Advanced Information Systems Engineering, Gdansk, Poland, 25–29 June 2012; Springer: Berlin/Heidelberg, Germany, 2012; pp. 238–253.
32. Corizzo, R.; Ceci, M.; Japkowicz, N. Anomaly detection and repair for accurate predictions in geo-distributed big data. *Big Data Res.* **2019**, *16*, 18–35. [CrossRef]
33. Sefidian, A.M.; Daneshpour, N. Estimating missing data using novel correlation maximization based methods. *Appl. Soft Comput.* **2020**, *91*, 106249. [CrossRef]
34. Rekatsinas, T.; Chu, X.; Ilyas, I.F.; Ré, C. HoloClean: Holistic Data Repairs with Probabilistic Inference. *Proc. VLDB Endow.* **2017**, *10*, 1190–1201. [CrossRef]
35. Yakout, M.; Elmagarmid, A.K.; Neville, J.; Ouzzani, M.; Ilyas, I.F. Guided Data Repair. *Proc. VLDB Endow.* **2011**, *4*, 279–289. [CrossRef]
36. Zhang, A.; Song, S.; Wang, J.; Yu, P.S. Time series data cleaning: From anomaly detection to anomaly repairing. *Proc. VLDB Endow.* **2017**, *10*, 1046–1057. [CrossRef]
37. Ge, C.; Gao, Y.; Miao, X.; Yao, B.; Wang, H. A Hybrid Data Cleaning Framework Using Markov Logic Networks. *IEEE Trans. Knowl. Data Eng.* **2020**. [CrossRef]
38. Chu, X.; Ilyas, I.F.; Krishnan, S.; Wang, J. Data cleaning: Overview and emerging challenges. In Proceedings of the 2016 International Conference on Management of Data, San Francisco, CA, USA, 26 June 2016; pp. 2201–2206.
39. Wang, X.; Wang, C. Time Series Data Cleaning: A Survey. *IEEE Access* **2019**, *8*, 1866–1881. [CrossRef]

40. Sadeghianasl, S.; ter Hofstede, A.H.; Wynn, M.T.; Suriadi, S. A contextual approach to detecting synonymous and polluted activity labels in process event logs. In Proceedings of the OTM Confederated International Conferences "On the Move to Meaningful Internet Systems", Rhodes, Greece, 21–25 October 2019; Springer: Cham, Switzerland, 2019; pp. 76–94.
41. Andrews, R.; Suriadi, S.; Ouyang, C.; Poppe, E. Towards event log querying for data quality. In Proceedings of the OTM Confederated International Conferences "On the Move to Meaningful Internet Systems", Valletta, Malta, 22–26 October 2018; Springer: Cham, Switzerland, 2018; pp. 116–134.
42. Kherbouche, M.O.; Laga, N.; Masse, P.A. Towards a better assessment of event logs quality. In Proceedings of the 2016 IEEE Symposium Series on Computational Intelligence (SSCI), Athens, Greece, 6–9 December 2016; pp. 1–8.
43. Lu, X.; Fahland, D. A Conceptual Framework for Understanding Event Data Quality for Behavior Analysis. In Proceedings of the 9th Central European Workshop on Services and their Composition Zeus Workshop 2017, Lugano, Switzerland, 13–14 February 2017; pp. 11–14.
44. Andrews, R.; van Dun, C.G.; Wynn, M.T.; Kratsch, W.; Röglinger, M.; ter Hofstede, A.H. Quality-informed semi-automated event log generation for process mining. *Decis. Support Syst.* **2020**, *132*, 113265. [CrossRef]
45. Sani, M.F. Preprocessing Event Data in Process Mining. In Proceedings of the 32nd International Conference on Advanced Information Systems Engineering, Grenoble, France, 8–12 June 2020; pp. 1–10.
46. Fox, F.; Aggarwal, V.R.; Whelton, H.; Johnson, O. A data quality framework for process mining of electronic health record data. In Proceedings of the 2018 IEEE International Conference on Healthcare Informatics (ICHI), New York, NY, USA, 4–7 June 2018; pp. 12–21.
47. Kurniati, A.P.; Rojas, E.; Hogg, D.; Hall, G.; Johnson, O.A. The assessment of data quality issues for process mining in healthcare using Medical Information Mart for Intensive Care III, a freely available e-health record database. *Health Inf. J.* **2019**, *25*, 1878–1893. [CrossRef] [PubMed]
48. Andrews, R.; Wynn, M.T.; Vallmuur, K.; Ter Hofstede, A.H.; Bosley, E.; Elcock, M.; Rashford, S. Leveraging data quality to better prepare for process mining: An approach illustrated through analysing road trauma pre-hospital retrieval and transport processes in Queensland. *Int. J. Environ. Res. Public Health* **2019**, *16*, 1138. [CrossRef]