IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 9 March 2024, accepted 2 May 2024, date of publication 6 May 2024, date of current version 17 May 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3397185

RESEARCH ARTICLE

Comparative Analysis of Classification-Based and Regression-Based Predictive Process Monitoring Models for Accurate and Time-Efficient Remaining Time Prediction

REZA AALIKHANI, MOHAMMAD FATHIAN, AND MOHAMMAD REZA RASOULI
School of Industrial Engineering, Iran University of Science and Technology, Narmak, Tehran 16846-13114, Iran
Corresponding author: Mohammad Fathian (fathian@iust.ac.ir)

ABSTRACT Predictive Process Monitoring (PPM) techniques leverage incomplete execution traces and historical event logs to predict outcomes, activities, and remaining time in ongoing processes. Accurately predicting process remaining time benefits process managers, enabling proactive decisions. A prediction model's effectiveness extends beyond accuracy, emphasizing timely predictions. Despite the continuous nature of time, the prevailing emphasis on regression-based approaches has overshadowed the untapped potential of classification-based methods. This study aims to perform a comparative analysis of Classification-Based PPM (CB-PPM) models and Regression-Based PPM (RB-PPM) models. The focus is on predicting remaining time in various processes, considering accuracy, offline execution time (model training), and online execution time (real-time predictions) as key evaluation criteria. We aim to assess the impact of model configuration on the performance of the prediction models. To accomplish this, our methodology includes designing experiments and implementing 136 PPM models on ten real-world datasets. These models configured with various combinations of four bucketing methods, five encoding methods, and eight prediction algorithms. The TOPSIS analysis highlights that the CB-PPM method is utilized in 90% of the most suitable models, whereas the RB-PPM method is found in only 10% of these models. The hypothesis testing results confirm that the CB-PPM method surpasses the RB-PPM method, significantly enhancing the accuracy of remaining time prediction. While the CB-PPM method has a higher online execution time, there is no observed increase in offline execution time. Furthermore, this study emphasizes the dataset-dependent nature of model configurations, underscoring that a single configuration may not universally apply to all datasets.

INDEX TERMS Predictive process monitoring, remaining time prediction, classification-based models, regression-based models, model configuration, comparative analysis.

I. INTRODUCTION

The discipline of Business Process Management (BPM) encompasses a range of methods for designing, implementing, measuring, controlling, and optimizing business processes [1, p. 16]. The emergence of process-aware information systems, such as Workflow Management (WFM) and Enterprise Resource Planning (ERP), has allowed for the storage of case execution information in event logs [2, p. 341]. Process mining is a related field that seeks to derive useful insights from these event logs [3]. Predictive Process Monitoring (PPM) is a more recent development within this domain, combining machine learning methods with process mining techniques to predict the future states of ongoing cases based on incomplete execution traces and historical event logs [2], [4, p. 419]. One key application of PPM is predicting the remaining time of cases in order to schedule processes effectively and allocate resources to cases

The associate editor coordinating the review of this manuscript and approving it for publication was Alba Amato.

at risk of deadline violations [3], [5], [6]. However, accurately predicting remaining time is a significant challenge in PPM, according to recent researches [7], [8].

Enhancing the accuracy of predicting the remaining time in a process not only improves the usability of the prediction results for process managers but also enables proactive decision-making [9]. However, the applicability of prediction models extends beyond accuracy alone, emphasizing the need for efficient computational time [10]. Therefore, it is vital to evaluate prediction models from both accuracy and computational time perspectives, considering their effectiveness across various processes. Despite the continuous nature of time, the predominant use of regression-based approaches for predicting remaining time has overshadowed the potential of classification-based methods in this domain [11], [12]. Hence, conducting a comprehensive examination of the advantages and disadvantages of the classification-based predictive process monitoring (CB-PPM) method in comparison to the regression-based predictive process monitoring (RB-PPM) method is essential for efficient prediction of remaining time in diverse processes.

Building upon these findings, this paper aims to explore the following research questions: (i) what are the advantages and disadvantages of different CB-PPM and RB-PPM models in terms of prediction accuracy, offline execution time (computational time for training the prediction model), and online execution time (computational time for making predictions for each running case) when predicting the remaining time of various business processes? (ii) which model, along with what configuration, is most suitable for accurately predicting time in different business processes? (iii) do CB-PPM and RB-PPM methods demonstrate advantages over each other in predicting remaining time? and (iv) does a prediction model with a specific configuration demonstrate suitability for predicting time in various business processes?

To comprehensively address the research questions, this study aims to evaluate a total of 136 PPM models on ten real-world datasets. It will involve configuring 68 CB-PPM models and 68 RB-PPM models, utilizing four bucketing methods, five encoding methods, and eight prediction algorithms. The evaluation will consider the accuracy, offline execution time, and online execution time of the models, both separately and in combination. The performance of the prediction models will be compared on each dataset individually based on each evaluation metric. Additionally, the study will analyze the effect of configuration on prediction performance by categorizing and comparing the model results based on different bucketing methods, encoding methods, and prediction algorithms. Statistical hypothesis tests will be employed to assess whether the selection of CB-PPM and RB-PPM methods yields statistically significant differences in accuracy, offline execution time, and online execution time of the prediction models. To obtain a comprehensive view, prediction models will be prioritized for each dataset by considering all evaluation metrics. Furthermore, the study will analyze the contribution of classification-based and regression-based methods in the first quartile of the top-performing models.

The rest of the paper is structured as follows. Section II presents the related works to this study, including an introduction to the PPM technique and its purpose in detail. Additionally, relevant studies on remaining time prediction are reviewed in this section. Section III represents the method used in this paper. This section includes the preliminary definitions of the PPM technique, a detailed description of the RB-PPM and CB-PPM methods, and a comprehensive presentation of the design of experiments methodology employed to conduct our experiments. Section IV represents the results of the implemented experiments on the CB-PPM and RB-PPM models. These results are compared based on accuracy, offline execution time, and online execution time metrics separately and simultaneously using TOPSIS methodology. Additionally, the results of hypothesis testing analysis are presented in this section. In Section V, we discuss the findings of the results and explore their practical implications. Additionally, the study's limitations are addressed in this section. Finally, Section VI summarizes the key findings and introduces future research directions.

II. RELATED WORKS

This section provides an introduction to the PPM technique and its purpose (see Subsection II-A). It then investigates relevant studies on remaining time prediction based on the research question addressed in this paper (see Subsection II-B). Through an exploration of relevant literature on this topic, this section aims to provide a comprehensive understanding of the different PPM concepts and their applications.

A. PPM

The field of predictive analytics involves using historical and current data to predict unknown future events. Within this domain, the PPM technique applies predictive analytics to business processes [13]. There are multiple PPM models available, and the optimal configuration for a given dataset may differ [14], [15]. Prior research in the PPM context can be categorized based on their prediction goals, methods used, predicted values, and prediction points. The following subsections (Subsection II-A1, II-A2, and II-A3) will elaborate on each of these categories.

1) PREDICTION GOALS

PPM models can be used for various prediction goals, such as predicting process performance criteria, predicting the violation of service level agreements (SLAs), predicting risk, and predicting the total outcome of the process. The time factor is considered as one of the most critical indicators of process performance [11], [16]. In addition, PPM models can predict other values such as the next events in the process or abnormal terminations [4], [11], [16], [17]. Past studies have often used statistical methods to predict the next events in the process [11].

2) PREDICTION METHODS

In PPM, prediction methods can be classified as non-process-aware and process-aware methods [7]. Non-process-aware methods do not utilize process model information in their predictions, while process-aware methods (e.g., Annotated Transition System (ATS)) incorporate a process model as input to their prediction procedure [3]. Process-aware predictive models are frequently used for time prediction in processes, such as predicting the total time of a process, the duration of an activity, or the remaining time in the process [11]. For example, a Naive Bayes (NB) predictive approach uses annotations on a labeled transition system (LTS) to predict the total time of the process and the sequence of the next activities [3].

3) PREDICTED VALUES

In PPM models, the predicted value can be either categorical or numerical, with the numerical values being further categorized as discrete or continuous [11]. Regression-based techniques are typically used to predict continuous numerical values, while classification-based techniques are employed to predict discrete and categorical values [3]. Moreover, accuracy indicators such as Root Mean Square Error (RMSE) are primarily used in regression-based techniques, whereas classification-based techniques rely on accuracy, precision, recall, and AUC as measures of accuracy [11].

B. REMAINING TIME PREDICTION USING PPM

The prediction of remaining time has become a significant challenge in process mining, as highlighted by recent studies [7], [8]. This prediction can be used to prevent deadline violations and ensure service-level objectives are met [15]. Prior studies have classified predicted times as remaining cycle time, delays, or remaining time in a business process [18], [19]. Specifically, the remaining time represents the time necessary to complete a process from a particular point in its execution [8]. Research in time prediction can be categorized as either process-aware or non-process-aware, as mentioned earlier. The following subsections (Subsection II-B1 and II-B2) provide an overview of existing time prediction studies, highlighting their approaches and outcomes.

1) PROCESS-AWARE REMAINING TIME PREDICTION METHODS

Process-aware methods for remaining time prediction typically utilize transition systems. One such study is [20], which uses a transition system annotated with the Naïve Bayes method to predict the time and sequence of the next activities. However, the study notes that this method is only effective for structured processes with limited activities. Another study, [21], clusters event logs based on case features and context features such as workload, and creates a transition model for each cluster to predict time. The remaining time of the process is then predicted through a multi-objective regression algorithm. In [22], a transition system annotated with the probabilities of a Fuzzy Support Vector Machine (FSVM) algorithm is used to determine the future path of a case, and the remaining time is calculated by adding up the predicted duration of the next activities using the Support Vector Regression (SVR) method. The study utilizes the shortest path algorithm to perform the prediction.

In addition to the aforementioned studies, researchers in [23] have used an annotated transition system (ATS) to predict the completion time of the process. This system has been improved by using an abstraction method that prevents over-fitting and under-fitting of the model. Also, this research has shown that the developed system is more accurate than regression-based and heuristic models. Similarly, in [8], the authors highlight that time prediction methods often do not consider all the information related to trace execution, such as the interval between activities, repetition of activities, or co-occurrences, which can reduce prediction accuracy. To address this, they propose a model that extracts structural features of traces, develops an ATS model that includes these features, and uses a partitioning strategy to categorize the features associated with each state in ATS. Finally, a linear regression-based predictor is used in each partition to predict the remaining time of new traces. Another innovative approach to predicting online cycle times in industrial environments is proposed in [9], where a product-oriented hybrid model is developed. This model is a weighted combination of a transition system and a statistical regression model, which is optimized with a linear programming model.

In addition to transition systems, various other models can be categorized as process-aware predictive models. For instance, researchers in [24] aim to predict the remaining process time by considering the elapsed time since the last observed event. To achieve this, they predict the total time of the process using the Petri net method. Unlike previous models that make a prediction only after a specific event has occurred, the developed model considers expected states that have not yet occurred in its predictions. However, relying solely on time statistics from historical cases is insufficient to produce reliable predictions. Therefore, both control and data flow perspectives are addressed in [25]. They utilize a process model augmented with time and data information to predict time. According to [26], a partial process model has been discovered using sequential pattern mining. By adding more information about activities to the partial model, predictions about the next step in the process and its expected duration can be made. The proposed model has been demonstrated to yield better results than annotated transition systems.

2) NON-PROCESS-AWARE REMAINING TIME PREDICTION METHODS

Non-process-aware methods have also been employed to predict remaining process time. For instance, in [27], the Bayesian neural network method is used in combination with the PPM technique to deal with uncertainties. By creating high-quality and low-quality answers, this method predicts uncertainty and generates a confidence interval, which is necessary due to noise in the data and uncertainty in the model. The authors also investigate the impact of data amounts and dataset size on prediction results. In [28], researchers highlight that although Deep Neural Networks (DNN) have been efficient in solving various problems, they are not superior to supervised algorithms for remaining time prediction. This is because categorical variables in tabular data resulting from trace encoding pose a challenge for the DNN method. To overcome this, the Entity Embedding technique has been developed to deal with tabular data that includes categorical variables.

In [29], authors utilized inter-case features for predicting remaining time. Specifically, they employed inter-case encoding based on case type, which enabled the partitioning of event logs into clusters with similar characteristics. This approach resulted in improved accuracy of the developed model, as indicated by the RMSE and Mean Absolute Error (MAE) accuracy metrics calculated in the study. Meanwhile, in [15], researchers employed the Long Short-Term Memory (LSTM) neural network method for predicting several outcomes, including the next event, its timestamp, and the remaining time. They noted that compared to other methods, the LSTM can be applied to a broader range of prediction goals without requiring specialized configurations.

In addition to the previously mentioned methods, some studies have adopted a hybrid approach. For instance, in [30], a deep learning model with a parallel structure, consisting of a Convolutional Neural Network (CNN) and a multi-layer perceptron model, is utilized to predict the remaining time of a process. The study's results indicate that the developed model exhibits a higher level of accuracy than the basic deep learning models. Additionally, in [16], the authors use a hybrid method. A clustering-based predictor is employed to predict the remaining time of an ongoing case, and a time-series model is utilized to predict future case results.

Based on the reviewed studies summarized in Table 1, it can be observed that the majority of methods used for predicting remaining time are process-aware. Non-process-aware methods have received less attention in the literature. Therefore, this study concentrates specifically on non-process-aware methods. Moreover, although the RB-PPM method has been extensively used for time prediction, the usage of the CB-PPM method has been limited. Previous studies investigating the RB-PPM method, such as [3], have primarily evaluated models based on single criteria such as accuracy, lacking a comprehensive perspective that considers multiple criteria simultaneously. To address these gaps and advance the understanding of time prediction using PPM, this study applies the CB-PPM method. Additionally, it conducts a comprehensive evaluation of the advantages and disadvantages of both CB-PPM and RB-PPM methods, utilizing various evaluation metrics.

III. MATERIAL AND METHODS

To comprehensively address the research questions and establish a robust methodological framework, our paper's methodology section is structured as follows. Subsection III-A offers an extensive overview of the preliminary definitions of the PPM technique, providing a solid foundation for the study. In Subsection III-B, we introduce the RB-PPM and CB-PPM methods, laying the groundwork for assessing their respective advantages and disadvantages. Building upon this foundation, Subsection III-C provides a detailed explanation of the design of experiments methodology employed to conduct our experiments, ensuring a systematic and rigorous approach to data analysis and comparison.

A. PRELIMINARY

Each entry in an event log is referred to as an "event" and represents the execution of a single work item in a process. These work items can be combined to form tasks in the process. A tuple $(a, c, t, r, (d_1, v_1), \ldots, (d_m, v_m))$ denotes an event, where 'a' represents the activity name, 'c' the case ID, 't' the timestamp of the event, 'r' the resource that performed the activity, and $(d_1, v_1), \ldots, (d_m, v_m)$ denotes other domain-specific attributes and their corresponding values. In this study, the sets $\mathcal{E}$, $\Re$, and Λ are used to represent the universe of all events, resources, and activities, respectively. Additionally, a dotted notation is used to refer to event attributes, for example, 'e.c' denotes the case ID of the event 'e'. A "trace" is a sequence of events created by a specific case, and it is represented by a non-empty sequence of events $\sigma = \langle e_1, \ldots, e_n \rangle$ where $\forall i \in [1..n], e_i \in \mathbb{E}$ and all of these events are associated with a specific case $(\forall i, j \in [1..n], e_i.c = e_j.c)$. In this study, the set $\mathcal{S}$ is used to represent the universe of all traces. The trace function is defined as $trace : \mathcal{E} \rightarrow \mathcal{S}$ which returns the trace associated with each event, e.g., $trace(e) = \sigma$. The attributes associated with a particular trace may be constant, referred to as case-level attributes, such as the case ID and requested amount in a loan application process. In contrast, event-level attributes are those whose values may change during a trace.

The prefix function is a function that returns the first l events of a trace σ (where $l < m$ and m is the length of σ), represented as $prefix(\sigma, l) = [e_1, \ldots, e_l]$. The prefix log of a log L is an event log that contains all prefixes of L, represented as, $L^* = \{prefix(\sigma, l) : \sigma \in L, 1 \leq l \leq |\sigma|\}$. A labeling function can be defined as $y : \mathcal{S} \rightarrow \mathcal{Y}$, which assigns a class label $y(\sigma) \in \mathcal{Y}$ to each trace $\sigma \in \mathcal{S}$, where $\mathcal{Y}$ is the domain of class labels. In the context of time prediction, $\mathcal{Y}$ is associated with a subset of natural numbers, $\mathcal{Y} = \{1, \ldots, N\}$ where $N \in \mathbb{N}$, and each number corresponds to a time interval. Since time is a continuous variable, it must be discretized to be predicted using a classification-based method. The equal-width discretization method is used to discretize the total time of cases. In this method, M is the number of intervals in which the data are categorized, and the width of each interval is calculated

TABLE 1. Summary of studies in the field of remaining time prediction using PPM approach.

		Characteristic		
Reference	**Prediction Model**	**Process-aware/Non-Process-Aware**	**Regression-based/ Classification-based**	**Goal**
[9]	A hybrid model of a Multilayer Regression and a Transition System	Process aware/ Non-process-aware	Regression based	Remaining Cycle Time
[30]	A hybrid Deep Learning	Non-process-aware	-	Remaining Time
[24]	A Petri Net model	Process aware	-	Remaining Service Execution Time
[20]	A hybrid data-aware Transition System and SVR	Process aware/ Non-process-aware	Regression based	Remaining Time/ Activities Sequence
[21]	A hybrid mode of Transition model and Multi-Target Regression	Process aware	Regression based	Processing Time/ SLA violation
[22]	A hybrid mode of Transition model and SVR	Process aware	Regression based	Remaining Time / Future Path
[25]	A Process model which is augmented by time and data information	Process aware	Regression based	Remaining Time
[26]	A hybrid model of a Partial Process model augmented by activities information and a Regression model	Process aware	Regression based	Completion Time/ Next Activity
[27]	Artificial Neural Networks	Non-process-aware	-	Remaining Time
[23]	Annotated Transition Systems	Process aware	-	Completion Time
[31]	Evolutionary Algorithms	Non-process-aware	-	Business Process Indicators
[7]	A multilayer hybrid model of LightGBM and XGBoost models	Non-process-aware	Regression based	Remaining Time
[8]	A hybrid model of Annotated Transition System and a Regression algorithm	Process aware	Regression based	Remaining Time
[28]	An augmented Deep Neural Network	Non-process-aware	-	Remaining Time
[15]	LSTM Neural Networks	Non-process-aware	-	Next Activity/Next Event Timestamp/ Remaining Time
[3]	Transition System/ Stochastic Petri Net(SPN)/ LSTM	Process aware/ Non-process-aware	Regression based	Remaining Time
[32]	LSTM	Non-process-aware	-	Remaining Trace and Runtime

as $Width = (maximum\ value{-}minimum\ value)/M$. After creating the intervals, events are labeled according to the total time of the case to which they belong. Some labels may be empty, so these empty labels should be deleted, and the remaining labels should be renumbered. The event labeling function $ye : \mathcal{E} \rightarrow \mathcal{Y}$, assigns a class label to each event in a trace. In remaining time prediction using the PPM technique, a predictor takes independent variables (features or attributes) as input data, trains on them, and predicts the dependent variable (total time for a running case). To use event log data in the training phase of prediction algorithms, each trace must be transformed into a fixed-length feature vector. This is accomplished using a trace encoder function $h : \mathcal{S} \rightarrow X_1 \times \cdots \times X_k$, which converts a trace into a K-dimensional feature vector $X_1 \times \cdots \times X_k$, where $1 \leq k \leq K$ and $X_k \in \mathbb{R}$ represents the domain of the K_{th} feature.

During encoding of a trace prefix into a feature vector, it is important to differentiate between inter-case and intra-case features. Intra-case features are related to a single case, such as the duration of this case or the number of events in this case. On the other hand, inter-case features are related to multiple cases in the process and can be categorized into resource-level and system-level features. The resource workload feature can be used as a representative of resource-level features that is number of tasks assigned to a specific resource. Furthermore, system-level features are related to the process as a whole, such as the number of active cases in the process. If a trace σ contains K features, R of these features are inter-case features, where R is less than or equal to K. The remaining $K - R$ features are considered as intra-case features. Additionally, K_1 of the R inter-case features are resource-level features, while K_2 of them are system-level features. It is important to carefully consider and select these features when encoding a trace into a feature vector for use in prediction algorithms.

The CB-PPM models utilize multi-class classification algorithms to predict time intervals (class labels) for a set of prefixes. These time intervals are associated with specific classes, where the desired time corresponds to the midpoint of these intervals. As the prediction algorithm outputs the process completion time for cases (total time of cases), the remaining time for each case can be derived by subtracting the elapsed time from the predicted total time.

B. CB-PPM AND RB-PPM METHODS

PPM techniques are widely employed for time prediction and follow a framework similar to Figure 1, as described by [4]. This framework consists of offline and online phases. The offline phase is dedicated to training a prediction model using the event log data. In contrast, during the online phase at runtime, the trained prediction model is utilized to make predictions for cases at the initial stages of the process. The PPM technique encompasses several components, which are detailed in the subsequent subsections. These components include event log preprocessing, prefix extraction and filtering, trace prefix bucketing, trace prefix encoding, and the prediction algorithm. The specific configuration of a prediction model is determined by the selection of the bucketing method, encoding method, and prediction algorithm.

The classification-based method and the regression-based method in the PPM technique differ primarily in the choice of prediction algorithm. The CB-PPM method utilizes a classification-based algorithm for its predictions, while the regression-based method employs a regression algorithm. In the case of time prediction, where time is a continuous variable, the classification-based method requires the discretization and labeling of the process completion time (total time) of cases. This discretization process is performed during the preprocessing step. Furthermore, when evaluating the accuracy metric for classification-based results, it is necessary to convert the predicted labels back to their corresponding time values, as explained in Subsection III-A. This conversion enables a meaningful comparison and assessment of the prediction accuracy.

1) EVENT LOG PREPROCESSING

To achieve high accuracy in PPM models, it is crucial to preprocess the input event logs [13]. Preprocessing begins with filtering or refining the dataset based on various criteria. For example, [4] separated event logs based on the total outcome of their cases and created different datasets. Feature engineering is then conducted to extract effective features for predicting the remaining time. These features can be categorized as time-related or case-related, with the latter being either intra-case or inter-case features. Outliers are also removed to prevent biasing the prediction results. In the context of time prediction, outliers are cases whose total process times differ significantly from those of other cases. Since time is a continuous variable, it needs to be discretized and labeled to use classification algorithms for time prediction. Several discretization methods exist, such as Equal Width Binning, Equal Frequency Binning, and Clustering-based methods [33]. In this study, we use the Equal Width Binning method because it is easy to understand and use, and useful for data visualization. This method divides the range of the continuous variable into equal-width intervals or bins, and the optimal number of bins is determined by the user during hyperparameter optimization. Overall, by carefully preprocessing the input event logs, including filtering and refining the dataset, feature engineering, outlier removal, and discretization and labeling of the cases' total times, we aim to achieve the highest level of accuracy in our PPM models. In Subsection III-C4, comprehensive information regarding the preprocessing steps applied to the chosen event logs is presented.

2) PREFIX EXTRACTION AND FILTERING

PPM models use a prefix log as input to train the predictor, which is a natural choice as we need to make predictions for partial traces at runtime instead of completed ones. Using a prefix log for training ensures that the training data is comparable to the testing data. For instance, for a complete trace with a total of five events, we can consider up to four prefixes, starting from the partial trace after executing the first event and incrementally adding events up to the fourth prefix. However, considering all possible prefixes can lead to computational constraints, making it necessary to reduce the number of created prefixes. To address this issue, some methods have been developed, such as gap-based filtering [34]. This approach retains prefixes whose length is equal to a base number plus a multiple of a gap, rather than retaining all prefixes up to a certain length. For example, for a gap of 5, the prefixes retained would be of length 1, 6, 11, 16, and 21. This method helps to keep the prefix log small enough for applications where computational efficiency is a significant concern.

3) TRACE PREFIX BUCKETING

PPM models use multiple classifiers instead of a single one to make predictions. To achieve this, the prefix traces in the historical log are divided into several buckets, where each bucket has different predictor trained on them. This bucketing is done to group cases with similar characteristics into the same bucket. During runtime, the most suitable bucket for the ongoing case is identified, and the corresponding predictor is used to make a prediction. This approach only considers cases that are most similar to the running case during prediction. There are different bucketing methods available, including single bucket, clustering-based, state-based, and prefix length-based bucketing. In the single bucket method, all prefix traces are considered to be in the same bucket, and a single classifier is trained on the entire prefix log. This single classifier is then applied directly to the running cases. The clustering-based method uses a clustering algorithm to determine the buckets (clusters) for the encoded prefix traces. One classifier is trained for each resulting cluster, considering only the historical prefix traces that fall into that particular cluster. At runtime, the cluster of the running case is determined based on its similarity to each of the existing clusters, and the respective classifier is applied. In the state-based bucketing approach, a process model is derived from the event log, and relevant states are determined from the process model. One classifier is trained for each state, and at runtime, the current state of the running case is determined. The respective classifier is then used to make a prediction for the running case. Finally, in the prefix length-based method, each bucket contains only the partial traces of a specific length. For example, one bucket contains traces where only the first event has been executed, another bucket contains those where the two first events have been executed, and so on. Then, one classifier is built for each possible prefix length (each bucket).

4) TRACE PREFIX ENCODING

In PPM, training a predictor requires representing all prefix traces within a bucket as fixed-length feature vectors. However, this is challenging because new information about the case is revealed with each executed event. Each trace in a bucket should still be represented by the same number of features, regardless of the number of events executed. To overcome this issue, a trace abstraction technique can be applied, such as only considering the last m events of the trace. However, selecting an appropriate abstraction can be difficult, as it requires balancing generality and information loss. Once a trace abstraction is chosen, a set of feature extraction functions can be applied to each attribute of the abstracted trace. Therefore, sequence encoding can be viewed as a combination of trace abstraction and feature extraction functions for each data attribute. Choosing the right abstraction and feature extraction functions is critical to ensure the accuracy of the predictor while minimizing the loss of relevant information. This process requires careful consideration and experimentation to identify the optimal combination of techniques.

There are multiple methods available for prefix encoding, which include last state, aggregation, index-based, and combined methods. For case attributes that contain categorical values, one-hot encoding is utilized across all methods. In this approach, each possible value of a categorical feature, which has n levels, is transformed into a bitvector $(v_1, \ldots, v_n)$. If the value of the feature is equal to the i_{th} level, then $v_i = 1, v_j = 0\ j \neq i$. The last state encoding method utilizes the most recent information of each prefix to encode the dynamic attributes, while disregarding previous information. This results in a fixed-size feature vector that is proportional to the number of event attributes and remains constant during case execution. However, this approach has the drawback of ignoring all previous data and only using the most recent snapshot. In contrast, the previous encoding method utilizes the pre-last state to encode the features. To encode prefixes using the aggregation method, all information in a trace is utilized, by integrating attribute values of all events in the trace using aggregation functions such as sum, minimum, maximum, and average. However, this approach ignores the order in which the events occurred, which can lead to loss of important information. The index-based encoding method is a powerful technique that takes into account all the information contained within a trace, including the order in which events occur. This makes it possible to convert the feature vector back into the original trace. In this method, a single feature is assigned to each event in the trace, resulting in a feature vector of varying lengths. However, it should be noted that this encoding method is only suitable for use with prefix length-based bucketing, as the different lengths of the feature vectors would make it difficult to use with other bucketing methods. The combined method is a hybrid encoding method that combines the benefits of the last state and aggregation methods. In this approach, both the last event and the frequency of events are considered, allowing for a more comprehensive representation of the trace.

5) PREDICTION ALGORITHM

To ensure accurate predictions in the online phase, it is essential to train a predictor for each bucket during the offline phase. This is accomplished by employing a classification-based or regression-based prediction algorithm. Indeed, classification-based algorithms necessitate discrete labels for prediction. Consequently, the total time for cases must be discretized and labeled accordingly to enable the application of these algorithms effectively. However, this process results in more than two labels, making it unsuitable for single-class classification algorithms. To address this challenge, multi-class classification algorithms such as Logistic Regression classifier (Logit-CL), Support Vector Machine (SVM), eXtreme Gradient Boosting classifier (XGBoost-CL), and Random Forest classifier (RF-CL) can be employed to configure CB-PPM models. Moreover, to configure RB-PPM models, regression algorithms like Decision Tree, Support Vector Regressor (SVR), eXtreme Gradient Boosting Regressor (XGBoost-RG), and Random Forest Regressor (RF-RG), can be utilized. The overall framework of the CB-PPM and RB-PPM methods is illustrated in Figure 1.

C. DESIGN OF EXPERIMENTS

Design of experiments (DOE) is a powerful methodology utilized in scientific research to systematically investigate the effects of different factors on a response variable [35, p. 13]. In this study, we employ the principles of DOE to compare the performance of two PPM methods: CB-PPM and RB-PPM. Through randomization of the PPM model configurations and conducting multiple replications on diverse datasets, our aim is to reduce biases, account for variability, and acquire reliable estimates of the methods' performance. By employing this robust experimental design, our research seeks to provide insightful findings on the relative effectiveness of the algorithms and contribute to the advancement of the field. The following paragraphs outline the key steps involved in the DOE method, highlighting our systematic approach to this comparative study. The following subsections (Subsection III-C1 to III-C6) represent details of the implemented DOE method in this study.

1) DEFINE THE FACTORS AND DETERMINE THE LEVELS

In this study, we define and determine the levels of four factors that are expected to impact the response variable. These factors were carefully selected based on their relevance

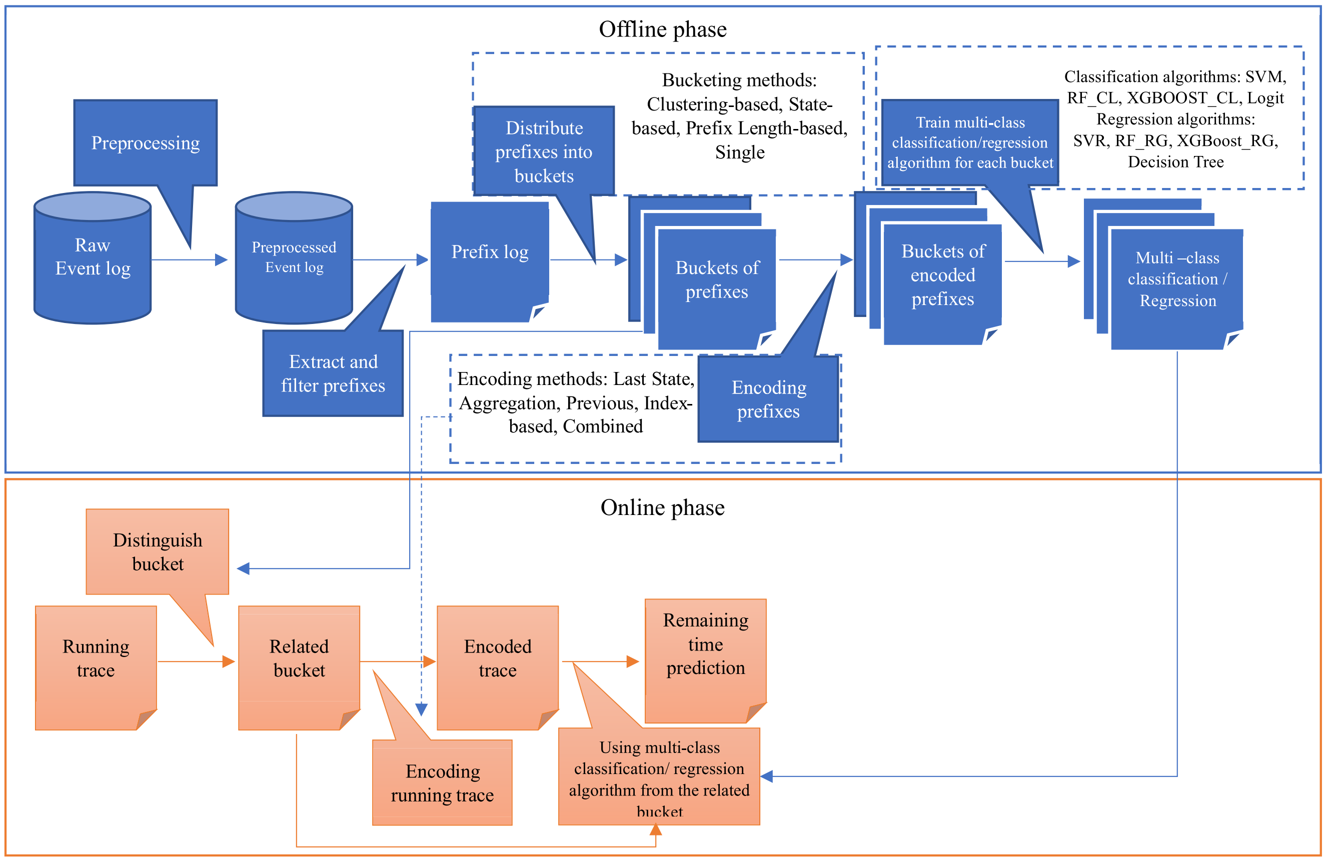
Offline phase
Preprocessing
Raw Event log
Preprocessed Event log
Extract and filter prefixes
Prefix log
Distribute prefixes into buckets
Bucketing methods: Clustering-based, State-based, Prefix Length-based, Single
Buckets of prefixes
Encoding methods: Last State, Aggregation, Previous, Index-based, Combined
Encoding prefixes
Buckets of encoded prefixes
Train multi-class classification/regression algorithm for each bucket
Classification algorithms: SVM, RF_CL, XGBOOST_CL, Logit
Regression algorithms: SVR, RF_RG, XGBoost_RG, Decision Tree
Multi –class classification / Regression
Online phase
Distinguish bucket
Running trace
Related bucket
Encoding running trace
Encoded trace
Using multi-class classification/ regression algorithm from the related bucket
Remaining time prediction

FIGURE 1. Overall framework of the PPM.

and potential influence on the outcome. They include the bucketing method, encoding method, prediction algorithm, and business process context. Figure 2 illustrates the factors associated with the configuration of prediction models (Factor 1 to Factor 3) and demonstrates how they are combined. The details of these factors are described as follows:

1. Bucketing method: In this factor, we examine various bucketing methods, including the single bucket, clustering-based, state-based, and prefix length-based bucketing methods.

2. Encoding Method: In this factor, we investigate various methods for data encoding. The methods under consideration include the last state, aggregation, index-based, and combined encoding methods.

3. Prediction Algorithm: In this factor, we explore different algorithms for prediction based on the CB-PPM and RB-PPM methods. For models based on CB-PPM, we consider the SVM, Logit-CL, XGBoost-CL, and RF-CL algorithms. Additionally, for RB-PPM based models, we include the SVR, Decision Tree, XGBoost-RG, and RF-RG algorithms.

4. Business process context: The context of the business process has a substantial influence on the response variables. Therefore, in this study, we have selected data from three distinct processes to capture this variability.

2) IDENTIFY THE RESPONSE VARIABLE

In this study, the performance of PPM models is comprehensively evaluated using three key criteria: accuracy, offline execution time, and online execution time. These criteria are widely employed in the field of time prediction and provide valuable insights into the effectiveness and efficiency of the models [4]. In practical applications, the capability of a PPM model to offer accurate predictions as early as possible from the process start time is important. The computational time required for generating predictions is an important factor, particularly for businesses that require fast response times [36]. By considering these metrics, we aim to provide insights into the effectiveness and efficiency of the models, enabling informed decision-making in practical applications. Further details on these evaluation metrics are presented below.

a: ACCURACY

To evaluate the accuracy of PPM models that predict continuous values, MAE and RMSE are commonly used metrics [11]. RMSE computes the square root of the average of the squared prediction errors, while MAE calculates the arithmetic mean of prediction errors. As RMSE heavily weighs significant errors by squaring them before averaging, it is sensitive to outlier values. On the other hand, MAE

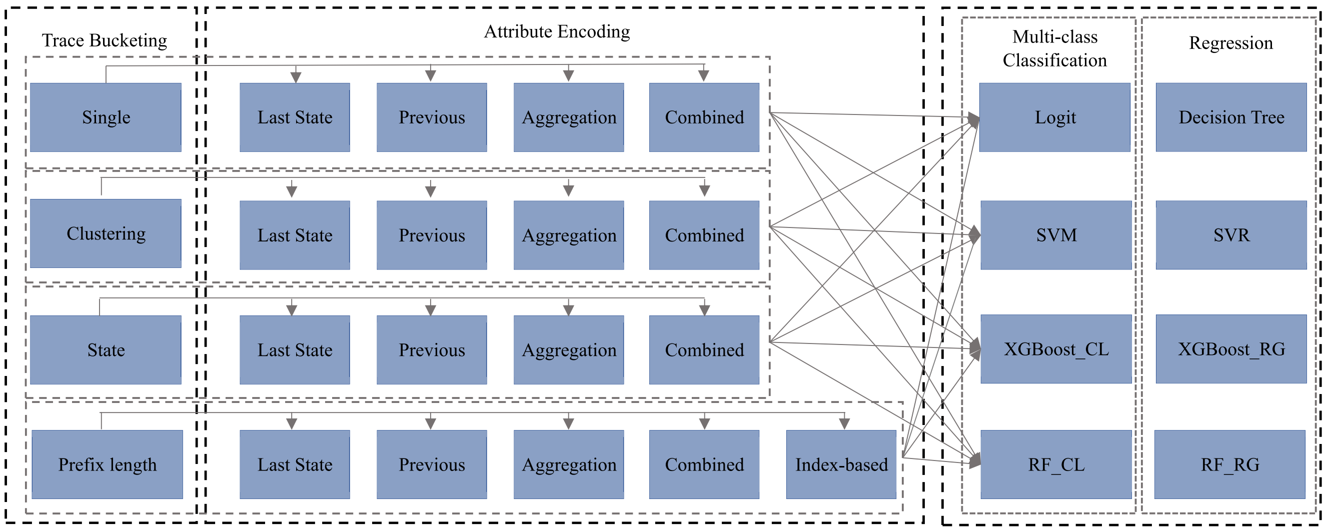

FIGURE 2. **Different configuration of the CB-PPM models and RB-PPM models, utilizing various bucketing methods, encoding methods, regression algorithms, and classification algorithms.**

measures errors only by their magnitude, without considering their direction, which makes it more suitable for handling outlier values. Since the remaining time of cases in a process can vary widely, we opt for the MAE metric to assess accuracy, as it is less affected by outliers. We use formula (1) to calculate the MAE metric, where $y_i \in \mathcal{Y} = \mathbb{R}$ represents the actual value of the total time for each case, and $\hat{y}_i \in \mathcal{Y} = \mathbb{R}$ is the predicted value (the predicted time value is the center of the time interval to which the predicted label belongs).

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |y_i - \hat{y}_i| \tag{1}$$

b: OFFLINE AND ONLINE EXECUTION TIMES

To evaluate PPM models based on computational time, we consider both offline and online execution times. Offline execution time refers to the time taken to construct a predictor from the trace data in the event log. This includes creating the prefix log, generating trace buckets, performing sequence encoding, and training the predictor. On the other hand, online execution time involves processing information following an upcoming event until a prediction can be made. This includes bucketing, encoding, and prediction based on the occurrence of a new event. It is important to note that the experiments were conducted using Python 3.10 and the scikit-learn library, and the hardware used was a four-core Intel(R) Core (TM) i7-7700k CPU @ 4.20 GHz with 16 GB of RAM.

3) PLAN THE EXPERIMENTAL DESIGN

To examine the impact of choosing CB-PPM and RB-PPM methods on prediction accuracy, offline execution time, and online execution time for each dataset, a hypothesis test is conducted to compare the average differences between these methods. The sample size required for the hypothesis test is determined using a power analysis method [37]. Based on formula (2), the effect size needs to be determined beforehand. In this study, a conservative estimate of effect size is adopted, assuming a smaller effect size than expected, which necessitates a larger sample size. Considering a moderate effect size of 0.5 (Cohen's d = 0.5), a desired power of 0.8, and a type 1 error rate of 0.05, a minimum of 51 samples for each method is determined. Therefore, at least 51 models (configurations) need to be run on each dataset. For this research, a total of 68 models (configurations) were considered for each CB-PPM and RB-PPM methods.

One of the research objectives is to investigate the differences in average accuracy, offline execution time, and online execution time between RB-PPM and CB-PPM methods while considering specific types of bucketing, encoding, or prediction algorithms. By fixing each of these factors, there are 16 different models. According to the power analysis method, a minimum of 51 samples is required to conduct the hypothesis test, necessitating the running of the models on at least four datasets. To enhance reliability, ten datasets are utilized for conducting the experiments. All models are implemented on all datasets to address the research question regarding the advantages and disadvantages of different models on different datasets. However, to perform the hypothesis test and incorporate randomization, the required number of samples is selected randomly from the experiments, ensuring that the influencing factors are randomly assigned. Additionally, to account for replication, the selected models are run again on the chosen datasets.

By providing a clear and detailed plan for the experimental design, including considerations for sample size determination, the number of models to be run, randomization, and replication, this section ensures the rigor and reliability of the study's findings.

$$n = \frac{2 \times (Z_{1-\alpha} + Z_{1-\beta})^2}{\mathit{effect\ size}^2} \tag{2}$$

4) CONDUCT THE EXPERIMENTS

This subsection presents the implementation details of the experiments conducted to assess the effectiveness of the CB-PPM and RB-PPM methods. The selected datasets and their preprocessing steps are described first, followed by a description of the experimental setup, which includes data splitting and hyperparameter optimization.

a: DATASET SELECTION

Previous studies in the context of PPM have commonly used real-life event logs to assess the performance of predictive models [3]. The Business Process Intelligence Challenge (BPIC) series, which contains real-life event logs, has been frequently used in PPM studies [3], [7], [8], [15], [38], [39]. In this study, we utilize the BPIC 2011, BPIC 2015, and BPIC 2017 datasets to evaluate the effectiveness of the CB-PPM method in comparison to the RB-PPM method. These datasets are publicly available at the 4TU[1] Center for Research Data. The BPIC 2017 dataset contains the historical event log of the loan application process at a Dutch financial institution. Cases in this dataset correspond to registered loan applications, and the outcome of each case is dependent on whether the application is accepted, canceled, returned, or refused [4]. To reduce computational costs, we consider cases with distinct request outcomes (BPIC17-Accepted, BPIC17-Canceled, BPIC17-Refused, and BPIC17-Returned) as separate datasets. The BPIC 2015 dataset consists of building permit application event logs from five Dutch municipalities, which are labeled as BPIC15-i, where i = 1…5 represents the related municipality. We treat each dataset related to a specific municipality as a distinct event log and apply a shared labeling function to all of them. Finally, the BPIC 2011 dataset contains traces from the Gynaecology department of a Dutch Academic Hospital. Each treatment is recorded as an activity associated with a given patient's medical history in this dataset.

b: PREPROCESSING OF SELECTED EVENT LOGS

The preprocessing steps are taken to prepare the datasets for analysis. The first step involves filtering cases based on their outcome to generate the BPIC14-Accepted, BPIC17-Canceled, BPIC17-Refused, and BPIC17-Returned datasets from the BPIC 2017 dataset (see Figure 3). The remaining datasets do not undergo any changes at this step. In the first step of feature engineering, time-related features are extracted, including the hour, weekday, time elapsed since the previous event, time elapsed since midnight, and time elapsed since the case started. To prevent biasing the prediction results, outliers are removed using Tukey's fences method. This involves calculating the interquartile range (IQR) and defining upper and lower fences as the third quartile plus 1.5 times the IQR and the first quartile minus 1.5 times the IQR, respectively. Any data point that falls outside these fences is considered an outlier. The total time of cases is discretized and labeled using the equal-width method with twenty bunches. The optimal number of bunches is calculated in the hyperparameter optimization subsection. Resource-related features are extracted in the second part of the feature engineering process. These features are calculated using resource multitasking and open cases. The "open cases" feature refers to the active cases per unit of time that have not yet been exited from the process, while the "resource workload" feature refers to the number of simultaneous services each resource provides for different cases. Tables 2 and 3 provide detailed information about the preprocessed event logs.

[1] https://data.4tu.nl/repository/collection:event_logs_real.

c: DATA SPLITTING

To simulate real-world conditions, a temporal split method is used in this study to divide the event log data into training and test sets. This method, as proposed by [4], involves using a separation time tag (t_s) to separate cases with a start time before t_s (i.e., $e_1.t < t_s$) into the training set, and cases with a start time after t_s into the test set. The data is split with a 20%/80% ratio, and cases with a start time before t_s but ending after t_s (i.e., $\exists i \mid e_i \in \sigma, e_i.t > t_s$) are removed to avoid using training data in the testing phase.

d: HYPERPARAMETER OPTIMIZATION

To utilize classification algorithms as predictors in CB-PPM models, it is necessary to discretize the total time of cases during the preprocessing phase. Since the number of labels during the discretization phase is often greater than two, multi-class classification algorithms should be used instead of Single-class ones. In this study, we employed RF-CL, XGBoost-CL, Logit, and SVM multi-class classification algorithms as predictors in the CB-PPM models for remaining time predictions. As described earlier in the Subsection III-A, we used the equal-width discretization method to discretize the total time of cases. However, determining the number of bunches (NOB) during the implementation of the equal-width method can be challenging. To identify the optimal number of bunches, we applied the grid search optimization method to the search space $NOB \in \{5, 10, 15, 20, 25, 30, 35\}$. For this purpose, we conducted the time prediction process for a model (with a configuration consisting of the prefix length-based bucketing method, the state-based encoding method, and the RF-CL classifier) considering the different number of bunches in the search space. The results revealed that the model achieved the highest accuracy when the number of bunches was set to 10. Therefore, we set the optimal number of bunches to 10 for all datasets.

In predictive modeling, both parameters and hyperparameters play a crucial role. While the parameters are learned during the training phase to fit the data, hyperparameters are tuned outside the training phase to control the learning cycle and achieve the best performance of the predictive models [38]. To optimize hyperparameters, the Tree-structured

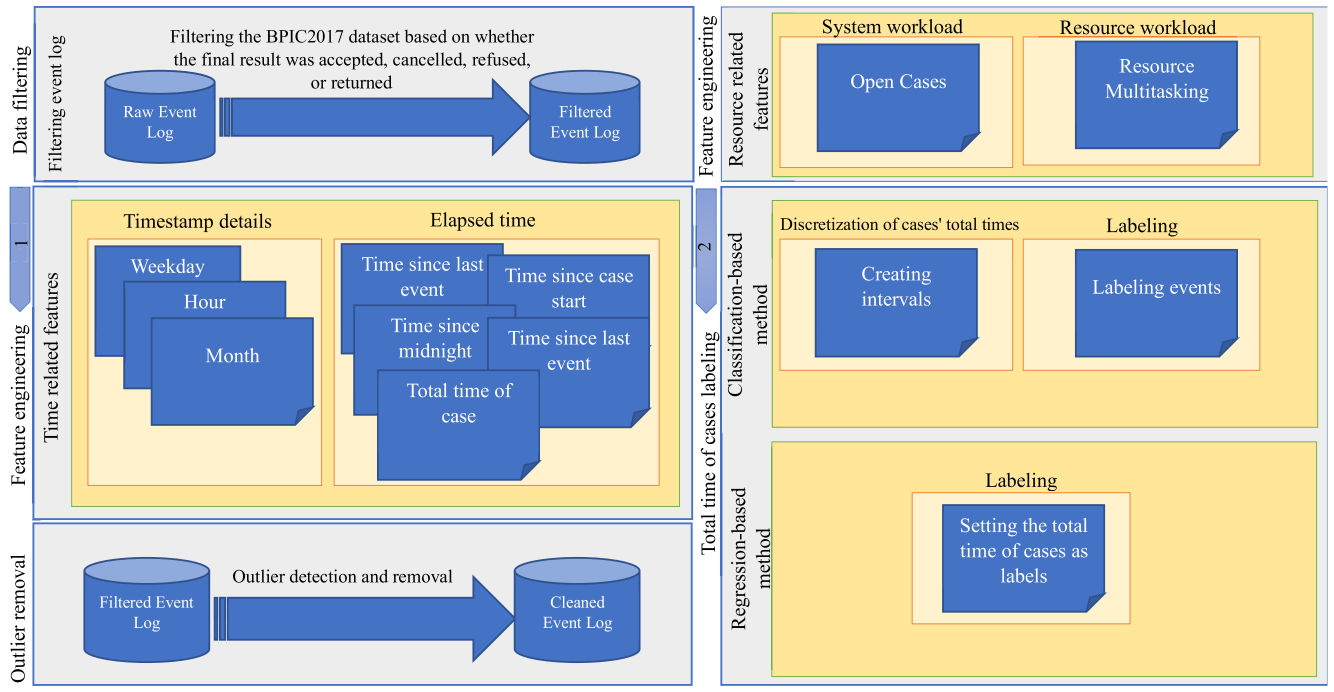

FIGURE 3. Overview of data preprocessing.

TABLE 2. Details of the preprocessed event logs.

	Dataset				
Features	**BPIC11**	**BPIC15-1**	**BPIC15-2**	**BPIC15-3**	**BPIC15-4**
min time stamp	2005-01-02 23:00:00	2010-10-04 22:00:00	2010-10-07 22:00:00	2010-10-03 22:00:00	2010-10-12 22:00:00
max time stamp	2008-03-19 23:00:00	2015-03-03 12:12:10	2015-02-27 08:47:51	2015-03-04 13:02:58	2015-03-01 23:00:00
Max number of events in traces	1814	80	132	124	79
Min number of events in traces	1	1	1	1	1
Average number of events in traces	222	23	30	23	23
Number of traces	1143	630	714	1242	552
Number of resources	67	20	11	13	8
Number of events	150291	24967	38319	52157	23075
Min duration of cases (s)	1440	513.65	939	3.75	720
Max duration of cases (s)	1664640	288744.2833	605598.6167	229691.95	392277.3167
Max number of multitasking	257	22	41	21	28
Average number of multitasking	20	4	10	5	5
Max number of open cases in the system	539	63	113	56	71
Number of dynamic categorical features	5	4	4	4	4
Number of static categorical features	6	2	2	2	2
Number of dynamic numerical features	4	10	10	10	10
Number of static numerical features	3	3	3	3	3

Parzen Estimator (TPE) method is used for each model separately [4]. One critical factor in applying the TPE method is the number of Optimization Rounds (OPR). Increasing the OPR improves the chances of obtaining globally optimal hyperparameter values but increases computational time. In this study, the OPR value for all models is set to 15. Finally, the hyperparameters and their search space for each prediction algorithm used in this study are presented in Table 4.

5) ANALYZE THE DATA

The experimental results are rigorously evaluated using three key evaluation metrics: accuracy, offline execution time, and online execution time. Each metric is analyzed independently to gain specific insights into the performance of the prediction models. Additionally, a comprehensive assessment is conducted by simultaneously considering all three metrics, allowing for a more holistic evaluation of the models' capabilities. During the evaluation, we identify the models with

TABLE 3. Details of the preprocessed event logs(continued).

	Dataset				
Features	**BPIC15-5**	**BPIC17-Accepted**	**BPIC17-Canceled**	**BPIC17-Refused**	**BPIC17-Returned**
min time stamp	2010-10-03 22:00:00	2016-01-01 09:51:15	2016-01-01 12:34:53	2016-01-01 10:16:11	2016-11-29 17:44:57
max time stamp	2015-02-26 11:51:27	2017-01-31 09:12:20	2017-02-01 10:15:04	2017-02-01 10:46:32	2017-02-01 14:11:03
max number of events in traces	101	129	159	149	108
min number of events in traces	1	1	1	1	1
average number of events in traces	27	23	22	22	28
number of traces	966	12764	14680	3704	25
number of resources	19	148	147	143	67
number of events	48827	531124	505399	143436	1282
min duration of cases (s)	1440	7.489133333	3.596216667	3.351033333	22328.24325
max duration of cases (s)	302460	87037.00005	87155.96572	86554.19343	85908.64345
max number of multitasking	50	21	597	13	2
average number of multitasking	6	1	10	1	1
max number of open cases in the system	98	966	1434	239	25
number of dynamic categorical features	4	6	6	6	6
number of static categorical features	2	3	3	3	3
number of dynamic numerical features	10	15	15	15	15
number of static numerical features	3	3	3	3	3

TABLE 4. Hyperparameters and distribution used for optimization using the TPE method.

Method	**Algorithm**	**Hyperparameter**	**Distribution**	**Values**
Classification	XGBoost-CL	Subsample	Uniform	$x \in [0.5, 1]$
		Learning rate	Uniform	$x \in [0, 1]$
		Colsample bytree	Uniform	$x \in [0.5, 1]$
		Max tree depth	Uniform integer	$x \in [4, 30]$
	RF-CL	Max features	Uniform	$x \in [0, 1]$
	SVM	Penalty parameter (C)	Uniform integer	$2x, x \in [-15, 15]$
		Kernel coefficient (gamma)	Uniform integer	$2x, x \in [-15, 15]$
	Logit	Inverse of regularization strength (C)	Uniform integer	$2x, x \in [-15, 15]$
Regression	XGBoost-RG	Subsample	Uniform	$x \in [0.5, 1]$
		Learning rate	Uniform	$x \in [0, 1]$
		Colsample bytree	Uniform	$x \in [0.5, 1]$
		Max tree depth	Uniform integer	$x \in [4, 30]$
	RF-RG	Max features	Uniform	$x \in [0, 1]$
	SVR	Penalty parameter (C)	Uniform integer	$2x, x \in [-15, 15]$
		Kernel coefficient (gamma)	Uniform integer	$2x, x \in [-15, 15]$
	Decision Tree	Max depth	Uniform integer	$x \in [1, 15]$
		Min samples leaf	Uniform integer	$x \in [1, 10]$
		Min weight fraction leaf	Uniform	$x \in [0, 0.5]$
		Max leaf nodes	Uniform integer	$x \in [10, 100]$

the best and worst performance on each dataset, considering each evaluation criterion independently. Furthermore, we categorize the results based on factors like bucketing methods, encoding methods, and prediction algorithms, investigating their impact on overall model performance. we categorize the results of the models based on whether they are derived from the classification or regression approach. The evaluation facilitated a comprehensive comparison of the performance between the CB-PPM and RB-PPM methods on individual datasets. This systematic evaluation approach enhances the credibility of our findings, providing valuable insights into the models' strengths and weaknesses under various conditions.

To holistically evaluate the prediction models, we employ the TOPSIS method, a powerful approach for multi-criteria decision-making problems (Please refer to [40] for more information regarding this method). The TOPSIS method facilitates ranking the models based on accuracy, offline execution time, and online execution time simultaneously, enabling a comprehensive assessment of their performance. Moreover, identifying models that excel across all three criteria is essential for making informed decisions. Additionally, to compare the advantages of the CB-PPM and RB-PPM methods, we determine the share of each method among the top 25% ranked models. This comparative analysis sheds light on their relative performance and aids in selecting the most effective approach for time prediction. Utilizing the TOPSIS method and conducting the comparative analysis ensures a robust and reliable evaluation, providing valuable insights for advancing predictive modeling in our research domain.

Moreover, to assess the statistical significance of the differences between the averages of accuracy, offline execution time, and online execution time for the RB-PPM and CB-PPM methods at a significance level of 95%, we employ hypothesis tests. Additionally, we investigate the significance of the difference between the averages of accuracy, offline execution time, and online execution time for the RB-PPM and CB-PPM methods while fixing one of the influencing factors, such as bucketing method, encoding method, or the examined dataset. This controlled analysis allows us to isolate the individual impact of each influencing factor on the performance metrics, providing valuable insights into the relative strengths and weaknesses of the RB-PPM and CB-PPM methods under different experimental conditions. By combining hypothesis testing and controlled comparisons, we enhance the validity and reliability of our findings, supporting informed decision-making and contributing to the advancement of predictive modeling in our research domain.

6) DRAW CONCLUSIONS AND MAKE RECOMMENDATION

Through a comprehensive evaluation of the results using individual evaluation metrics, we gain valuable insights into the performance of different models and their configurations for time prediction on each dataset. This analysis helps us identify models with the highest accuracy and the shortest offline and online execution times consistently across all datasets. Additionally, we assess the influence of different factors, such as bucketing method, encoding method, and prediction algorithm, on the evaluation criteria, providing crucial insights into their effectiveness under varying conditions. Based on these conclusions, we can select the most appropriate bucketing method, encoding method, and prediction algorithm for each dataset to optimize accuracy and computational efficiency.

Furthermore, employing the TOPSIS method allows us to identify the best models that exhibit superior performance, characterized by reduced offline and online execution times and high accuracy. Additionally, when considering all three evaluation criteria simultaneously, the method highlights the advantages of using the CB-PPM and RB-PPM approaches. These findings aid in drawing conclusions about the effectiveness of specific configurations in addressing various datasets, facilitating informed decision-making in real-world applications.

Moreover, the results of the hypothesis test offer valuable insights into the advantages of the CB-PPM and RB-PPM approaches relative to each other in terms of accuracy improvement and reduction in offline and online time, as observed overall. By fixing each influencing factor, including the bucketing method, encoding method, and used dataset, we can determine the specific advantages of these methods for time prediction. This comprehensive analysis allows us to draw conclusions about the overall superiority of each approach and their performance under different influencing factors. These findings deepen our understanding of the strengths of the CB-PPM and RB-PPM approaches, providing valuable guidance for their optimal selection in various time prediction scenarios.

Overall, our systematic evaluations and conclusions provide essential recommendations for optimizing model configurations and selecting the most suitable techniques for accurate and efficient time prediction in practical applications. These insights contribute to advancing the field of predictive modeling and enhancing decision-making in real-world contexts.

IV. RESULTS

This section presents the results of the implemented experiments on the CB-PPM and RB-PPM models. The comparison of these models is based on three evaluation metrics: MAE values, offline execution time, and online execution time, which are separately presented in different subsections (Subsection IV-A, IV-B, and IV-C, respectively). Each subsection identifies the models that perform the best and worst based on each evaluation criterion and provides insights into the most suitable bucketing method, encoding method, and prediction algorithm. Additionally, the performance of the CB-PPM and RB-PPM methods is compared across each dataset. Subsection IV-D details the outcomes of the TOPSIS method (with a weight of 0.9 for the MAE metric and a weight of 0.1 for online and offline times), where the best and least-performing models considering all evaluation metrics are identified. The performance of CB-PPM and RB-PPM models is also compared using this method. Lastly, the results of the hypothesis testing are presented in Subsection IV-E, evaluating the superiority of CB-PPM over RB-PPM both overall and under the scenario of holding one of the influencing factors constant, such as the bucketing method, encoding method, or used dataset. These comprehensive analyses provide valuable insights into the strengths and weaknesses of the CB-PPM and RB-PPM methods and their performance in various evaluation scenarios.

In this section, it is essential to clarify that the sequences, such as Clustering-Last State-RF-CL, serve as model names. These names indicate the specific configuration of each model, where Clustering represents the bucketing method being clustering-based, Last State signifies the usage of the last state encoding method, RF denotes the Random Forest prediction algorithm, and CL indicates the PPM method being classification-based. In the case of applying the RB-PPM method, the model's name will be extended with the expression RG at the end, distinguishing it from the CB-PPM method.

A. MAE METRIC

Based on the MAE values of the models represented in Figure 4 to 13 in Appendix A, and the summarized outcomes in Table 5 and Table 6, it is evident that the configuration of the most accurate models varies across different datasets. However, the Single-Last State-XGBoost-CL model displayed superior accuracy in three datasets. Conversely, the Single-Last State-Decision Tree-RG and Prefix-Aggregation-SVM-CL models were identified as the least accurate models in two datasets each. Regarding the bucketing method perspective, the single bucket method in the CB-PPM models was found to be the most suitable choice in 90% of the datasets. However, single bucket method in the RB-PPM models emerged as the worst-performing bucketing method in three datasets. Based on the evaluation of encoding methods, it was observed that the previous encoding method was the most suitable choice in 60% of the datasets in models based on the CB-PPM approach. However, in models employing the RB-PPM method, both previous encoding and index-based encoding methods were identified as the least performing encoding methods in 60% of the datasets. Regarding the prediction algorithms, the Random Forest classifier (RF-CL) demonstrated the highest accuracy in 70% of the datasets. Conversely, the XGBoost regressor (XGBoost-RG) exhibited the least accuracy in 40% of the datasets. Analyzing the performance of the CB-PPM and RB-PPM approaches, it was found that the CB-PPM approach outperformed the RB-PPM approach in 80% of the datasets, highlighting its superiority in time prediction tasks.

B. OFFLINE EXECUTION TIME METRIC

Based on the offline execution time of the models, as depicted in Figure 14 to 23 in Appendix B and summarized in Table 7 and Table 8, it is evident that the fastest models differ across datasets, indicating that specific configurations do not consistently yield the fastest performance across all datasets. However, the Single-Previous-Decision Tree-RG model emerged as the fastest model in three datasets. On the other hand, the State-Aggregation-XGBoost-RG and State-Last State-XGBoost-RG models were the slowest models in two datasets each. From the bucketing method perspective, the single bucket and prefix length-based bucketing methods in models based on the CB-PPM approach demonstrated the fastest offline performance in six datasets. Conversely, the state-based bucketing method in RB-PPM models showcased the slowest offline performance in five datasets. Regarding the encoding method perspective, the index-based encoding method in CB-PPM demonstrated the fastest offline phase performance (being the fastest in five datasets). Conversely, the last state, previous, and aggregation encoding methods in RB-PPM models displayed the slowest offline performance in two datasets each. From the prediction algorithm perspective, the Decision Tree regressor was the fastest prediction algorithm in 60% of datasets, while the XGBoost regressor (XGBoost-RG) was the slowest in 80% of datasets. Comparing the CB-PPM and RB-PPM approaches, the CB-PPM models were faster than the RB-PPM models in 80% of the datasets, while in the remaining datasets, the RB-PPM models were faster. Indeed, the results clearly demonstrate the superiority of the CB-PPM approach over the RB-PPM approach in terms of offline time for time prediction.

C. ONLINE EXECUTION TIME METRIC

Based on the online execution time of the models, as illustrated in Figure 24 to 33 in Appendix C and summarized in Table 9 and Table 10, no single model is consistently the fastest across all datasets. However, the Prefix-Previous-XGBoost-CL and Prefix-Last State-XGBoost-CL models emerged as the fastest in two datasets each. Conversely, the Clustering-Combined-RF-RG and State-Combined-SVM-CL models were the slowest in four datasets. From the bucketing method perspective, the prefix length-based bucketing method in CB-PPM models demonstrated the shortest online time in 50% of datasets. In contrast, the clustering-based bucketing method in RB-PPM models exhibited the slowest online performance in 40% of datasets. Considering the encoding method perspective, the index-based encoding method in CB-PPM models proved to be the fastest in four datasets. On the other hand, the combined encoding method in CB-PPM models displayed the slowest performance in four datasets. From the prediction algorithm perspective, both the XGBoost-RG and XGBoost-CL algorithms were the fastest in three datasets each. In contrast, the RF-CL algorithm had the slowest online time in five datasets. Overall, in 60% of datasets, the RB-PPM models demonstrated faster online performance compared to CB-PPM models, while in the remaining datasets, CB-PPM models exhibited faster online execution times. It reveals the superiority of RB-PPM approach over the CB-PPM approach in the online phase of time prediction.

D. TOPSIS ANALYSIS

Based on the TOPSIS analysis results, as summarized in Table 11, both the Prefix-Last State-RF-CL and Prefix-Previous-RF-CL models obtained the highest rank in three datasets each, indicating their superior overall performance. Conversely, the Clustering-Previous-XGBoost-RG model had the lowest rank in two datasets, reflecting its comparatively poorer performance. From a broader perspective, in 80% of the datasets, all models within the highest 25% ranks (first quartile of the best-ranked models) were based on the CB-PPM approach, suggesting that CB-PPM models tend to outperform RB-PPM models in the majority of scenarios.

E. HYPOTHESIS TESTING RESULTS

We conducted a hypothesis test to investigate whether the CB-PPM method leads to a statistically significant decrease in the average MAE for time prediction compared to the

TABLE 5. Summary of results based on the MAE metric.

	Dataset				
Features	**BPIC11**	**BPIC15-1**	**BPIC15-2**	**BPIC15-3**	**BPIC15-4**
Best model	Cluster-Aggregation-XGBoost -CL / 41.3 days	Single-Last State - XGBoost -CL / 12.4 days	Single-Last State - XGBoost -CL / 15.3 days	Single-Combined-RF-RG / 7.4 days	Prefix- Previous-XGBoost -CL / 13.4 days
Worst model	Single-Last State-Decision Tree-RG / 293.6 days	Single-Last State-Decision Tree-RG / 43.6 days	Single-Previous-XGBoost-RG / 98.1 days	Prefix-Aggregation-SVM-CL / 50.3 days	Prefix-Aggregation-XGBoost-RG / 114.8 days
Best bucketing method	Single bucket-CL / 84.1	Single bucket-CL / 18.2 days	Single bucket-CL / 28.2 days	Clustering based-RG / 11.7 days	Single bucket-CL / 18.3 days
Worst bucketing method	State based-RG / 244.1 days	State based-RG / 35.0 days	Single bucket-RG / 59.9 days	State based-CL / 22.5 days	Clustering based-RG / 84.7 days
Best encoding method	Previous-CL / 80.7 days	Last State-CL/ 20.9 days	Previous-CL / 29.3 days	Index based-RG / 12.0 days	Last State-CL/ 27.5 days
Worst encoding method	Index based-RG / 242.2 days	Index based-RG / 36.0 days	Previous-RG / 63.1 days	Aggregation-CL / 19.8 days	Aggregation-RG / 85.2 days
Best prediction algorithm	XGBoost-CL /52.1 days	RF-CL / 14.3 days	RF-CL / 18.5 days	RF-RG / 10.2 days	XGBoost-CL / 20.5 days
Worst prediction algorithm	Decision Tree-RG / 258.6 days	Decision Tree-RG / 35.3 days	XGBoost-RG / 63.1 days	SVM-CL / 21.4 days	XGBoost-RG / 85.4 days
MAE of CB-PPM on average	106.5 days	23.3 days	35.3 days	18.3 days	30.8 days
MAE of RB-PPM on average	234.6 days	34.1 days	54.7 days	12.3 days	81.1 days

TABLE 6. Summary of results based on the MAE metric(continued).

	Dataset				
Features	**BPIC15-5**	**BPIC17-Accepted**	**BPIC17-Canceled**	**BPIC17-Returned**	**BPIC17-Refused**
Best model	Prefix-Last State-XGBoost-CL / 13.0 days	Cluster- Previous-XGBoost -CL/ 1.4 days	Single-Aggregation-RF-CL / 1.37 days	Single-Last State-XGBoost-CL / 1.9 days	Single-Last State-RF-CL / 1.4 days
Worst model	Single-Combined-Logit-CL / 106.9 days	Cluster-Aggregation-SVM -CL / 7.7 days	Single-Aggregation-SVM-RG / 10.4 days	Single-Last State-Logit-CL / 20.8 days	Prefix-Aggregation-SVM-CL / 8.1 days
Best bucketing method	Single bucket-CL / 24.1 days	Single bucket-CL / 2.6 days	Single bucket-CL / 2.7 days	Single bucket-RG / 6.3 days	Single bucket-CL / 2.7 days
Worst bucketing method	Prefix Length based-RG / 24.9 days	Clustering based-RG / 7.1 days	Single bucket-RG / 8.9 days	State based-CL / 10.7 days	Single bucket-RG / 7.3 days
Best encoding method	Previous-CL / 22.8 days	Previous-CL / 2.8 days	Previous-CL / 3.0 days	Index based-RG / 6.6 days	Previous-CL / 2.8 days
Worst encoding method	Index based-CL / 38.8 days	Previous-RG / 7.1 days	Index based-RG / 9.0 days	Aggregation-CL / 10.0 days	Previous-RG / 7.2 days
Best prediction algorithm	RF-CL / 14.8 days	RF-CL / 1.4 days	RF-CL / 1.4 days	RF-CL / 4.6 days	RF-CL / 1.5 days
Worst prediction algorithm	SVM-CL / 41.1 days	XGBoost-RG / 7.3 days	XGBoost-RG / 9.3 days	Logit-CL / 14.7 days	Decision Tree-RG / 7.6 days
MAE of CB-PPM on average	27.5 days	3.2 days	3.3 days	8.6 days	3.3 days
MAE of RB-PPM on average	33.8 days	7.1 days	8.7 days	6.8 days	7.2 days

RB-PPM method. The null hypothesis (H0) states that there is no difference or an increase in the average MAE when using the CB-PPM method, while the alternative hypothesis (H1) posits that the average MAE is reduced with the CB-PPM method ($H0 : \mu_d \geq 0, H1 : \mu_d \leq 0$, where μ_d represents the mean difference between the two methods). The hypothesis testing was conducted for each dataset separately and for all datasets combined. The same approach was used to test whether using the CB-PPM method increased offline and online execution times. Furthermore, the results of the implemented models on all datasets were systematically categorized based on their bucketing methods, encoding methods, and prediction algorithms, followed by the execution of hypothesis testing for comprehensive analysis.

The results of the hypothesis testing conducted to investigate the superiority of the CB-PPM and RB-PPM methods in decreasing the MAE for time prediction demonstrated that, overall, the CB-PPM method is more effective in reducing the MAE value (as presented in Table 12).Out of the ten datasets analyzed, improvements in accuracy were proven in eight datasets when using the CB-PPM method, while two datasets did not show significant improvement. This indicates that the CB-PPM method outperforms the

TABLE 7. Summary of results based on the offline execution time evaluation metric.

	Dataset				
Features	**BPIC11**	**BPIC15-1**	**BPIC15-2**	**BPIC15-3**	**BPIC15-4**
fastest model	Single-Previous-Decision Tree-RG / 1.232 minutes	Single-Last State-Decision Tree-RG / 0.69 minutes	Single-Combined-SVM-RG / 1.33 minutes	Single-Previous-Decision Tree-RG / 0.69 minutes	Single-Last State-Decision Tree-RG / 1.57 minutes
slowest model	State-Previous-XGBoost-RG / 1089.3 minutes	State-Aggregation-XGBoost-RG / 862.6 minutes	State-Last State-XGBoost-RG / 1268.9 minutes	State-Last State-XGBoost-RG / 1630.4 minutes	State-Aggregation-XGBoost-RG / 1212.9 minutes
fastest bucketing method	Single bucket-CL / 46.1 minutes	Single bucket-CL / 19.8 minutes	Prefix Length based-RG / 34.5 minutes	Single bucket-CL / 29.7 minutes	Prefix Length based-CL / 24.4 minutes
slowest bucketing method	State based – RG / 309.9 minutes	State based – RG / 229.4 minutes	State based – RG / 301.7 minutes	Clustering based-RG / 394.0 minutes	Clustering based-RG / 355.7 minutes
fastest encoding method	Index based-RG / 53.3 minutes	Index based-CL / 33.0 minutes	Index based-CL / 44.2 minutes	Index based-CL / 24.9 minutes	Index based-CL / 27.1 minutes
slowest encoding method	Last State-RG / 207.0 minutes	Previous-RG / 159.0 minutes	Aggregation-RG / 301.6 minutes	Last State-RG / 335.9 minutes	Combined-RG / 274.5 minutes
fastest prediction algorithm	Decision Tree-RG / 4.9 minutes	SVM-CL / 6.1 minutes	SVM-CL / 8.2 minutes	SVM-CL / 15.9 minutes	SVM-CL / 12.5 minutes
slowest prediction algorithm	XGBoost-RG / 602.6 minutes	XGBoost-RG / 450.1 minutes	XGBoost-RG / 790.3 minutes	XGBoost-RG / 827.1 minutes	XGBoost-RG / 793.4 minutes
Offline time of CB-PPM on average	85.2 minutes	58.5 minutes	75.8 minutes	85.4 minutes	110.4 minutes
Offline time of RB-PPM on average	172.5 minutes	143.1 minutes	234.9 minutes	255.9 minutes	237.2 minutes

TABLE 8. Summary of results based on the offline execution time evaluation metric(continued).

	Dataset				
Features	**BPIC15-5**	**BPIC17-Accepted**	**BPIC17-Canceled**	**BPIC17-Returned**	**BPIC17-Refused**
fastest model	Single-Previous-Decision Tree-RG / 1.15 minutes	Prefix-Last State-Decision Tree-RG / 1.42 minutes	State-Previous-Decision Tree-RG / 3.6 minutes	Single-Combined-SVM-RG / 0.61 minutes	Single-Aggregation-Decision Tree-RG / 3.2 minutes
slowest model	State-Combined-XGBoost-CL / 1390.8 minutes	Clustering-Previous-XGBoost-RG / 3476.9 minutes	Prefix-Last State-XGBoost-RG / 3782.8 minutes	Clustering-Aggregation-XGBoost-CL / 251.1 minutes	Single-Aggregation-XGBoost-RG / 3479.4 minutes
fastest bucketing method	Prefix Length based-RG / 102.5 minutes	Prefix Length based-CL / 158.0 minutes	Prefix Length based-CL / 152.3 minutes	Single bucket-RG / 31.4 minutes	Prefix Length based-RG / 148.8 minutes
slowest bucketing method	State based – CL / 425.1 minutes	State based – RG / 555.8 minutes	Prefix Length based-RG / 915.8 minutes	Clustering based-CL / 77.2 minutes	State based – RG / 752.5 minutes
fastest encoding method	Index based-CL / 30.2 minutes	Aggregation-CL / 123.6 minutes	Aggregation-CL / 202.0 minutes	Aggregation-RG / 39.8 minutes	Index based-RG / 231.9 minutes
slowest encoding method	Last State-CL / 364.6 minutes	Aggregation-RG / 572.3 minutes	Index based-RG / 899.9 minutes	Aggregation-CL / 67.7 minutes	Previous-RG / 624.4 minutes
fastest prediction algorithm	Decision Tree-RG / 4.9 minutes	Decision Tree-RG / 7.8 minutes	Decision Tree-RG / 11.7 minutes	Decision Tree-RG / 3.9 minutes	Decision Tree-RG / 14.4 minutes
slowest prediction algorithm	XGBoost-CL / 1150.9 minutes	XGBoost-RG / 1620.7 minutes	XGBoost-RG / 2072.4 minutes	XGBoost-CL / 204.0 minutes	XGBoost-RG / 1501.1 minutes
Offline time of CB-PPM on average	341.4 minutes	160.6 minutes	215.8 minutes	64.1 minutes	404.1 minutes
Offline time of RB-PPM on average	186.6 minutes	434.2 minutes	576.5 minutes	42.4 minutes	425.5 minutes

RB-PPM method in accurately predicting the remaining time. Furthermore, accuracy improvements were observed while using the CB-PPM method in all scenarios of fixing bucketing methods. This finding suggests that the choice of bucketing method does not significantly impact the superiority of the CB-PPM approach in predicting time accurately. Moreover, when considering fixed encoding methods, it was observed that, with the exception of the index-based encoding method, the utilization of the CB-PPM approach consistently led to improved accuracy across various encoding scenarios. This finding highlights the significance of the encoding method in determining the superiority of the CB-PPM approach for

TABLE 9. Summary of results based on the online execution time evaluation metric.

	Dataset				
Features	BPIC11	BPIC15-1	BPIC15-2	BPIC15-3	BPIC15-4
fastest model	Clustering-Aggregation-SVM-RG /0.012 minutes	Prefix-Previous-XGBoost-CL / 0.004 minutes	Prefix-Last State-XGBoost-CL / 0.004 minutes	Prefix-Previous-XGBoost-CL / 0.004 minutes	Prefix-Previous-Logit-CL / 0.003 minutes
slowest model	Clustering-Combined-Logit-CL model /0.265 minutes	Clustering-Last State-Decision Tree-RG / 0.334 minutes	Clustering-Combined-RF-RG / 0.079 minutes	Single-Combined-RF-RG / 0.136 minutes	Clustering-Combined-RF-RG / 0.062 minutes
fastest bucketing method	prefix length based-CL /0.051 minutes	prefix length based-CL /0.012 minutes	prefix length based-CL /0.012 minutes	prefix length based-CL /0.011 minutes	prefix length based-CL /0.008 minutes
slowest bucketing method	Single bucket-CL /0.221 minutes	Clustering based-RG / 0.211 minutes	Clustering based-RG / 0.056 minutes	Clustering based-RG / 0.049 minutes	Clustering based-RG / 0.045 minutes
fastest encoding method	Index based-CL / 0.055 minutes	Index based-CL / 0.016 minutes	Last State-CL / 0.011 minutes	Last State-CL / 0.013 minutes	Index based-CL / 0.012 minutes
slowest encoding method	Combined-CL / 0.170 minutes	Last State-RG / 0.113 minutes	Combined-RG / 0.038 minutes	Combined-RG / 0.043 minutes	Combined-RG / 0.028 minutes
fastest prediction algorithm	XGBoost-RG /0.046 minutes	Logit-CL / 0.014 minutes	XGBoost-CL /0.010 minutes	Logit-CL / 0.014 minutes	XGBoost-CL /0.009 minutes
slowest prediction algorithm	RF-CL /0.174 minutes	Decision Tree-RG / 0.120 minutes	RF-RG /0.041 minutes	RF-RG /0.043 minutes	RF-RG /0.035 minutes
online time of CB-PPM on average	0.159 minutes	0.159 minutes	0.042 minutes	0.014 minutes	0.02 minutes
online time of RB-PPM on average	0.066 minutes	0.066 minutes	0.088 minutes	0.031 minutes	0.03 minutes

TABLE 10. Summary of results based on the online execution time evaluation metric(continued).

	Dataset				
Features	BPIC15-5	BPIC17-Accepted	BPIC17-Canceled	BPIC17-Returned	BPIC17-Refused
fastest model	Prefix-Previous-XGBoost-RG / 0.004 minutes	Prefix-Last State-Decision Tree-RG / 0.004 minutes	Clustering-Previous-XGBoost-RG / 0.007 minutes	Prefix-Last State-XGBoost-CL / 0.004 minutes	State-Last State-Decision Tree-RG / 0.009 minutes
slowest model	Clustering-Combined-RF-CL / 0.075 minutes	State-Combined-SVM-CL / 0.134 minutes	State-Combined-SVM-CL / 0.168 minutes	Clustering-Combined-Logit-CL / 0.063 minutes	State-Previous-RF-RG / 0.176 minutes
fastest bucketing method	prefix length based-RG /0.008 minutes	prefix length based-RG /0.016 minutes	Clustering based-RG / 0.019 minutes	prefix length based-RG /0.014 minutes	Clustering based-CL / 0.021minutes
slowest bucketing method	Clustering based-CL / 0.045 minutes	State based-CL / 0.082 minutes	State based-CL / 0.113 minutes	Clustering based-CL / 0.043 minutes	Single bucket-RG / 0.092 minutes
fastest encoding method	Index based-CL / 0.015 minutes	Last State-RG / 0.023 minutes	Last State-RG / 0.025 minutes	Last State-RG / 0.014 minutes	Last State-CL / 0.030 minutes
slowest encoding method	Combined-CL / 0.036 minutes	Index based-CL / 0.071 minutes	Combined-CL / 0.074 minutes	Combined-CL / 0.032 minutes	Index based-RG / 0.069 minutes
fastest prediction algorithm	XGBoost-CL /0.012 minutes	XGBoost-RG /0.021 minutes	Decision Tree-RG / 0.034 minutes	XGBoost-RG /0.008 minutes	SVM-CL / 0.046 minutes
slowest prediction algorithm	RF-CL /0.051 minutes	RF-CL /0.057 minutes	RF-CL /0.077 minutes	RF-CL /0.038 minutes	RF-RG /0.065 minutes
online time of CB-PPM on average	0.015 minutes	0.03 minutes	0.052 minutes	0.063 minutes	0.026 minutes
online time of RB-PPM on average	0.024 minutes	0.024 minutes	0.03 minutes	0.037 minutes	0.017 minutes

accurate time prediction. Notably, the index-based encoding method appears to be particularly effective for RB-PPM models.

The hypotheses tested to examine the effect of the CB-PPM method on offline execution time led to an overall conclusion that there was no evidence supporting an increase in offline execution time. However, the results of the hypothesis tests showed variations across datasets. Specifically, the hypothesis of increased offline execution time was supported in the datasets BPIC15-5, BPIC17-Returned, and BPIC17-Refused, but rejected in the remaining datasets. Interestingly, when fixing the bucketing method, the increase in offline time caused by applying the CB-PPM method was rejected in all scenarios. This suggests that the choice of bucketing method does not significantly impact the superiority of the CB-PPM approach in terms of computation time. On the other

TABLE 11. Summary of results based on TOPSIS analysis.

Features	Dataset									
	BPIC11	BPIC15-1	BPIC15-2	BPIC15-3	BPIC15-4	BPIC15-5	BPIC17-Accepted	BPIC17-Canceled	BPIC17-Returned	BPIC17-Refused
Best ranked model	Prefix-Last State-RF-CL	Prefix-Previous-RF-CL	Prefix-Last State-RF-CL	Single-Last State-SVM-RG	Prefix-Last State-XGBoost-CL	Prefix-Index-XGBoost-CL	Prefix-Previous-RF-CL	Prefix-Previous-RF-CL	Single-Last State-RF-CL	Prefix-Last State-RF-CL
Worst ranked model	Single-Last State-XGBoost-RG	Single-Last State-Decision Tree-RG	Single-Previous-XGBoost-RG	Prefix-Aggregation-SVM-CL	Prefix-Aggregation-XGBoost-RG	Single-Combined-Logit-CL	Clustering-Previous-XGBoost-RG	Clustering-Previous-XGBoost-RG	State-Aggregation-SVM-CL	Single-Aggregation-XGBoost-RG
CB-PPM precent of first quartile	100%	100%	100%	18%	100%	100%	100%	100%	62%	100%
RB-PPM precent of first quartile	0%	0%	0%	82%	0%	0%	0%	0%	38%	0%

hand, fixing the encoding method led to the rejection of the hypothesis that CB-PPM would increase offline time in all scenarios except for the combined and index-based encoding methods. This finding implies that the choice of encoding method significantly influences superiority of the CB-PPM approach in achieving time-efficient predictions during the offline phase. Finally, the results of the hypothesis testing indicate that the CB-PPM method did not increase the offline execution time of prediction, despite its ability to improve accuracy.

The results of the hypothesis testing conducted to investigate to comparing the CB-PPM and RB-PPM methods on online execution time showed that the CB-PPM method generally increases online execution time compared to the RB-PPM method. However, there were some datasets (BPIC15-1 to BPIC15-4) where an increase in online time was not observed. This suggests that while using the CB-PPM approach in time prediction may result in increased online execution time in general, there might be exceptions for certain datasets. In the scenarios where bucketing and encoding methods were fixed, the CB-PPM method demonstrated an increase in online execution time. As a result, the choice of bucketing or encoding methods cannot negate the superiority of the RB-PPM approach in terms of online execution time. So, while the CB-PPM method can improve accuracy, it may also come with the tradeoff of increased online execution time.

V. DISCUSSION

As explained in the hypothesis test section, the accuracy of models based on the CB-PPM method is generally higher than models based on the RB-PP method. By considering the most accurate models conducted on all datasets, it is noticeable that the single-bucket bucketing method was employed in 69% of these models. This result reveals that separating prefixes based on different criteria like prefix length or states that the prefixes refer to can't significantly affect the accuracy of models. All of the encoding methods were utilized in the most accurate models but the previous and last state encoding methods were used more than others and index-based encoding was used less than others (27%, 27%, and 3%, respectively). So, the last state or the last two states in which the case was placed have the greatest impact on accurately predicting the remaining time of the process. A significant finding was that 99% of the most accurate models were based on the CB-PPM method, while only 1% used the RB-PPM method. This result of the study reveals that selecting an appropriate configuration of the model based on the CB-PPM approach can lead to more accurate predictions across different datasets.

According to the hypothesis test, using the CB-PPM method did not result in a significant increase in offline execution time. However, the use of single-bucket bucketing was observed in 80% of the fastest models, indicating that separating prefixes based on their characteristics can increase the offline execution time of models. The previous encoding method was the most frequently used encoding method among the fastest models (40%), as it only considers information from the last two states and encode them into the feature vectors. It's worth noting that all of the fastest models were based on the RB-PPM method. Although the CB-PPM method improves accuracy, it may not be effective in suggesting models that are quickest in the offline phase.

According to the hypothesis test results, CB-PPM models had a longer average online time than RB-PPM models. However, when considering the fastest models implemented on different datasets, 70% of them used prefix length-based bucketing. This indicates that bucketing prefixes based on their length can speed up online execution time, as it facilitates finding the bucket corresponding to the running case. In terms of encoding, the previous encoding method was the most commonly used in the fastest models (50%). Interestingly, the fastest models were equally based on the CB-PPM method and the RB-PPM method, indicating that both methods can contribute equally to generating models that are fastest in the online phase.

TABLE 12. Hypothesis testing of improving accuracy, increasing offline total time, and increasing online average time by considering the CB-PPM method than RB-PPM method.

		MAE related hypothesis			Offline time related hypothesis			Online time related hypothesis		
Category	Filter	t-value	p-value	hypothesis	t-value	p-value	hypothesis	t-value	p-value	hypothesis
Bucketing method	Clustering-based	3.7262188	0.0001216	H1	2.1937421	0.0145668	H1	0.8190746	0.2066791	H0
	State-based	2.6516762	0.0042557	H1	2.5324627	0.0060176	H1	-3.5824034	0.0002059	H0
	Single bucket	4.7261607	0.000002	H1	2.2434372	0.0129495	H1	-2.5563008	0.0055549	H0
	Prefix length-based	4.0057556	0.0000388	H1	1.8344084	0.0338584	H1	0.1896343	0.4248467	H0
Encoding method	Aggregation	3.396331	0.0003989	H1	2.4660162	0.0072164	H1	-1.5170293	0.0652057	H0
	Last state	4.3547268	0.0000104	H1	1.936998	0.0269797	H1	-0.670702	0.2514486	H0
	Combined	2.6886896	0.0038088	H1	1.6477056	0.0503473	H0	-1.7191925	0.0433347	H0
	Previous	4.5387177	0.0000047	H1	2.304478	0.0110638	H1	-0.802395	0.2114663	H0
	Index-based	1.379114	0.0861949	H0	1.3648505	0.0891962	H0	-0.3810907	0.3521559	H0
Dataset	BPIC11	15.6755327	0	H1	2.1888358	0.0158619	H1	-8.7602608	0	H0
	BPIC15-1	8.4339352	0	H1	2.9114635	0.0022055	H1	3.3580158	0.0005534	H1
	BPIC15-2	7.0682669	0	H1	3.16255	0.0011107	H1	5.5248655	0.0000002	H1
	BPIC15-3	-6.0772652	0	H0	2.9898512	0.0018996	H1	2.3162101	0.0110919	H1
	BPIC15-4	21.2183852	0	H1	2.4704591	0.0076237	H1	3.843956	0.0001025	H1
	BPIC15-5	2.7541104	0.003722	H1	-2.2561245	0.0131024	H0	-1.8044438	0.0369937	H0
	BPIC17-Accepted	15.9497416	0	H1	2.5568932	0.0063378	H1	-4.7289583	0.0000034	H0
	BPIC17-Canceled	20.743362	0	H1	2.8398133	0.002897	H1	-3.7141528	0.0001681	H0
	BPIC17-Returned	-2.9258318	0.002291	H0	-1.7695542	0.0397151	H0	-3.7642957	0.0001411	H0
	BPIC17-Refused	15.2594181	0	H1	0.1722	0.4318105	H0	1.2716014	0.1028957	H0
	all	7.5010881	0	H1	4.3973691	0.0000061	H1	-2.3561461	0.0093069	H0

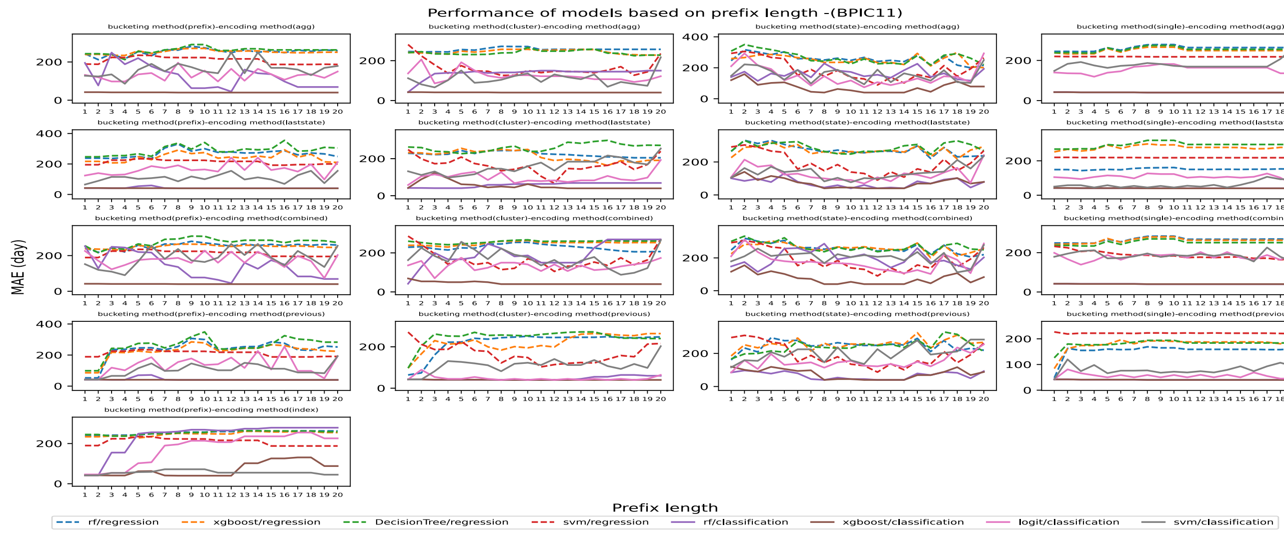

FIGURE 4. MAE of PPM models on BPIC11 dataset by prefix length, categorized by bucketing and encoding methods.

The findings discussed in the previous paragraphs were based on evaluating the metrics separately. However, when all evaluation metrics were simultaneously considered using TOPSIS analysis, it was found that 80% of the most suitable models used prefix length-based bucketing as their bucketing method. This suggests that this bucketing method improves models' accuracy without increasing offline and online execution times. Regarding the encoding methods, 60% of the most efficient models used the last state encoding method. This finding indicates that although the last state encoding method only considers information from the last state of the running case and ignores other information from its trace, it improves the accuracy of models and speeds up offline and online execution times. The reduction in effectiveness of models by considering more information of a trace may be due to the fact that considering all trace information adds noise and non-useful information to the feature vector, which reduces prediction accuracy. Additionally, considering more information from a trace increases the data required for processing, which increases offline and

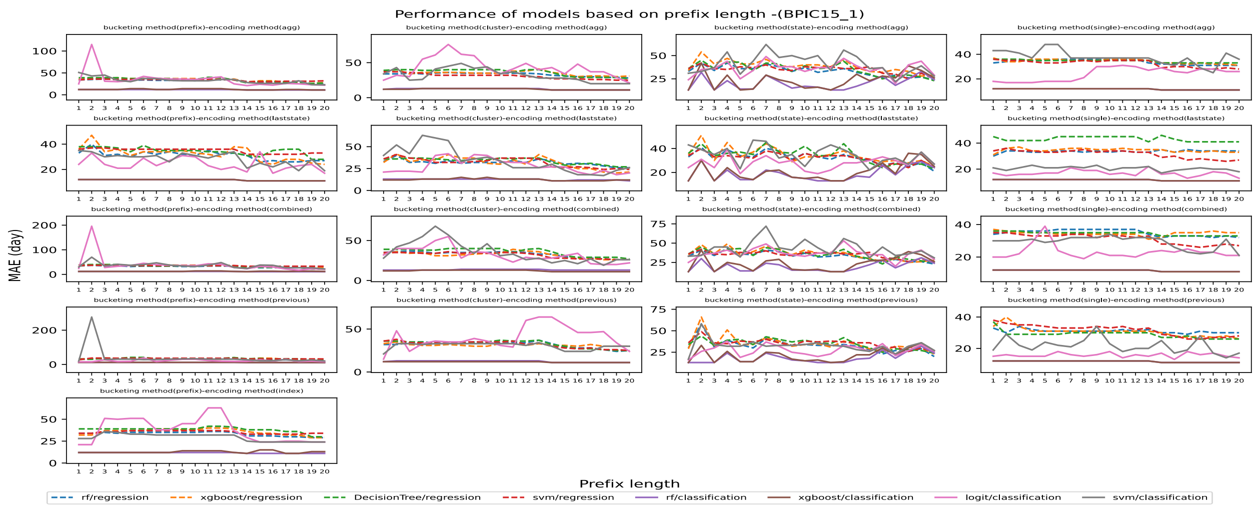

FIGURE 5. MAE of PPM models on BPIC15-1 dataset by prefix length, categorized by bucketing and encoding methods.

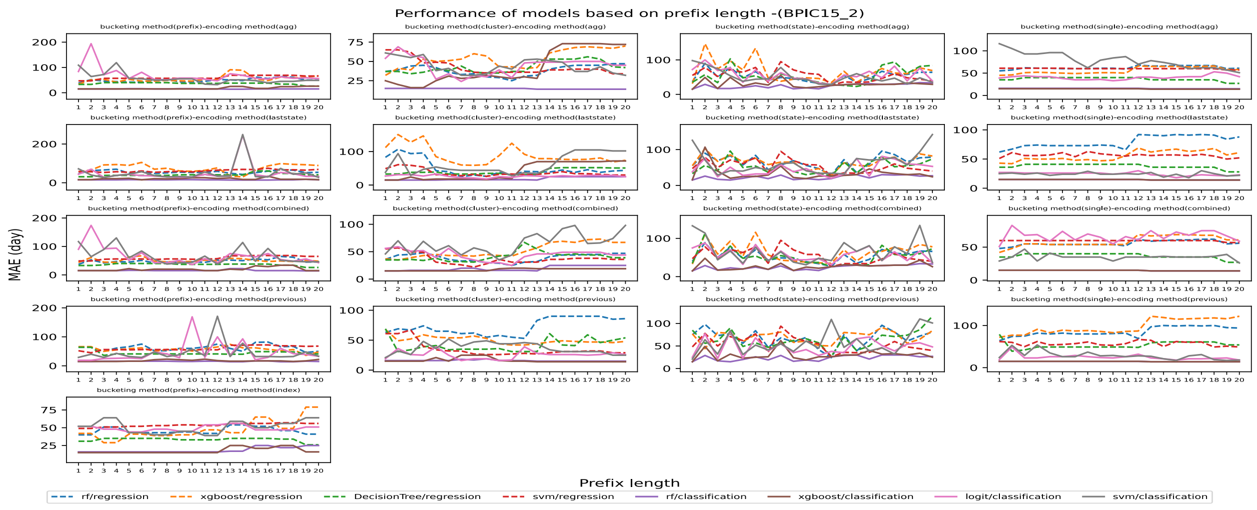

FIGURE 6. MAE of PPM models on BPIC15-2 dataset by prefix length, categorized by bucketing and encoding methods.

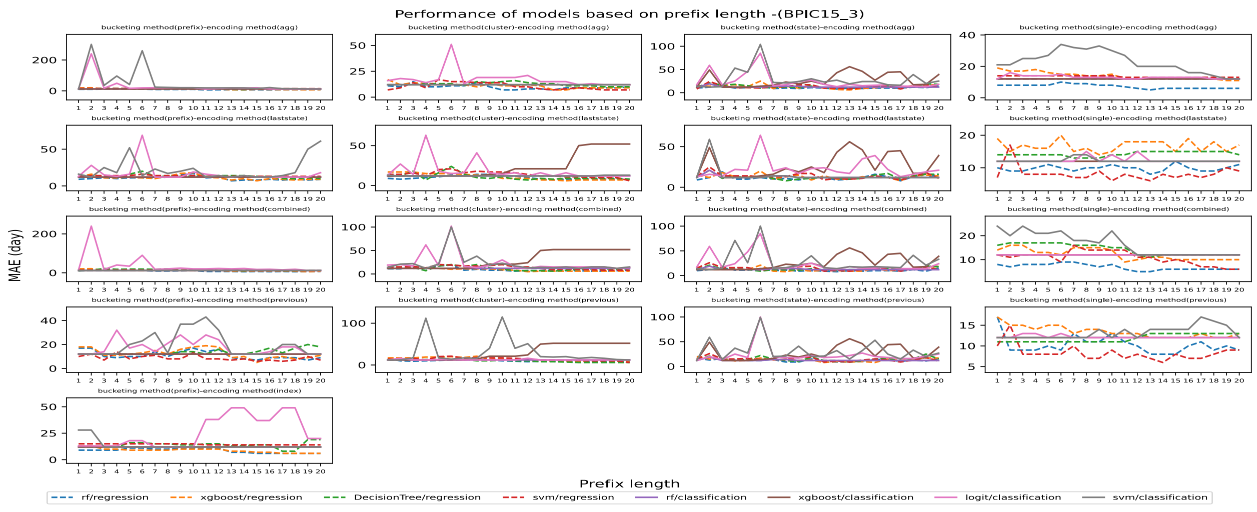

FIGURE 7. MAE of PPM models on BPIC15-3 dataset by prefix length, categorized by bucketing and encoding methods.

online processing time and reduces the effectiveness of the model. An important finding of the result analysis is that 90% of the most suitable models were based on the CB-PPM method, while only 10% used the RB-PPM method.

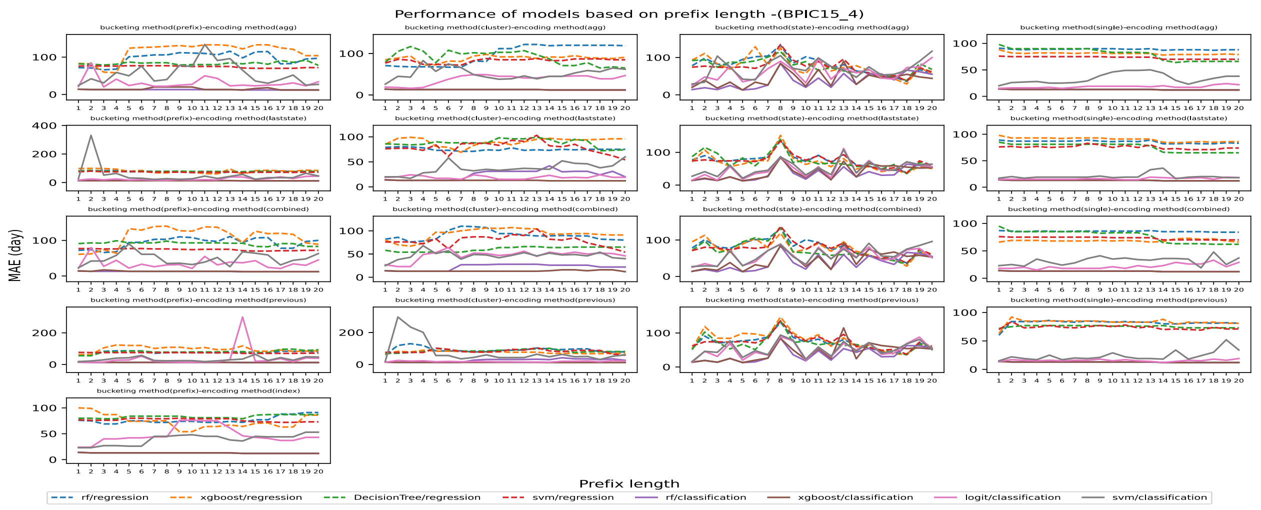

FIGURE 8. MAE of PPM models on BPIC15-4 dataset by prefix length, categorized by bucketing and encoding methods.

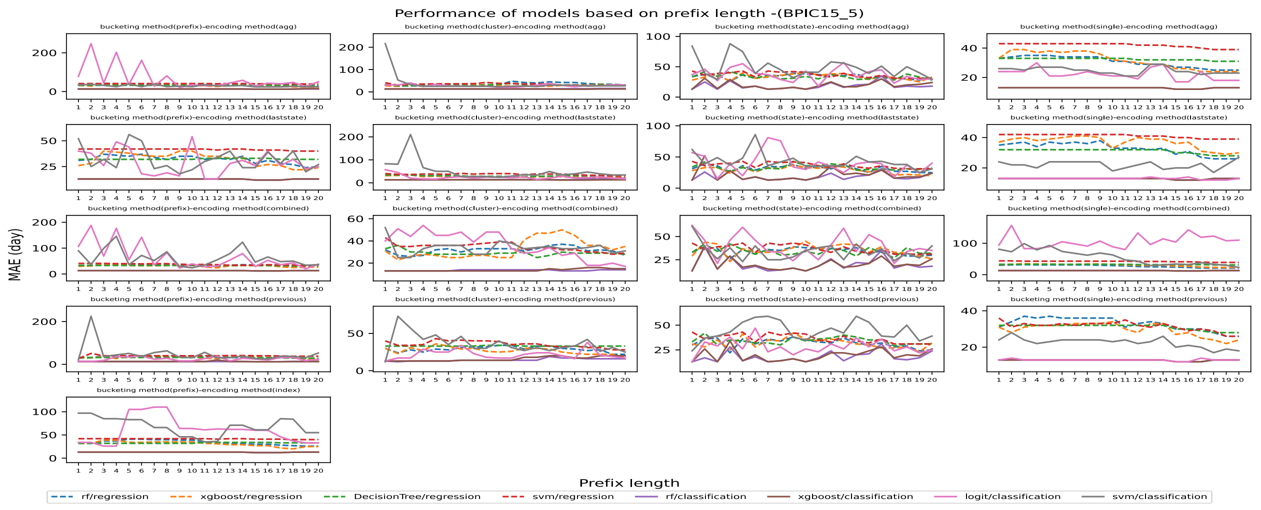

FIGURE 9. MAE of PPM models on BPIC15-5 dataset by prefix length, categorized by bucketing and encoding methods.

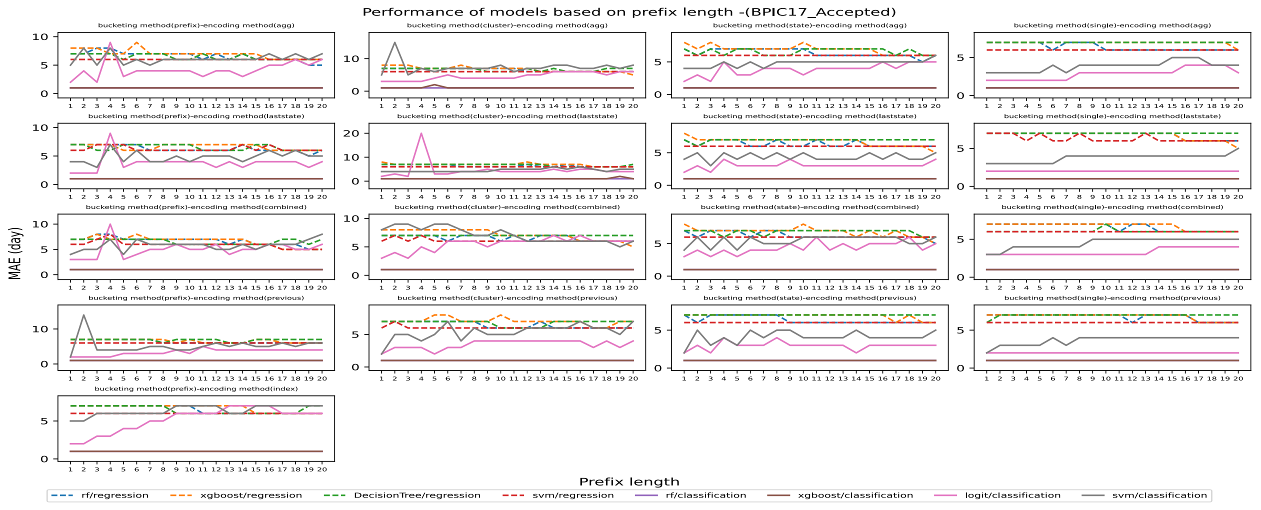

FIGURE 10. MAE of PPM models on BPIC17-Accepted dataset by prefix length, categorized by bucketing and encoding methods.

This indicates that the CB-PPM method outperforms the RB-PPM method in time prediction considering accuracy and execution time of models simultaneously. Furthermore, 70% of the most suitable models used RF classification

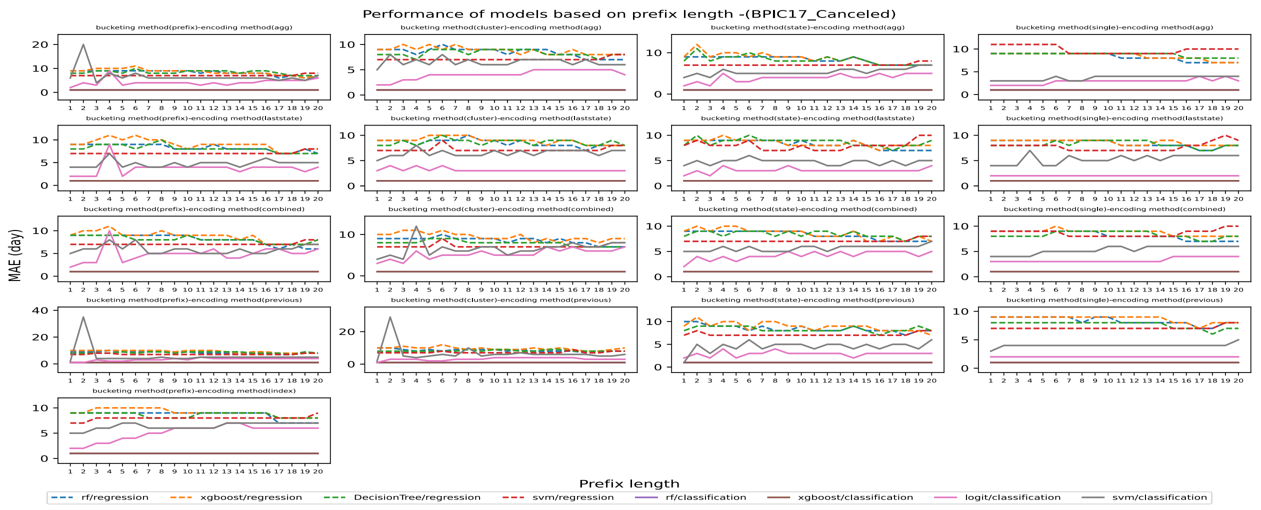

FIGURE 11. MAE of PPM models on BPIC17-Canceled dataset by prefix length, categorized by bucketing and encoding methods.

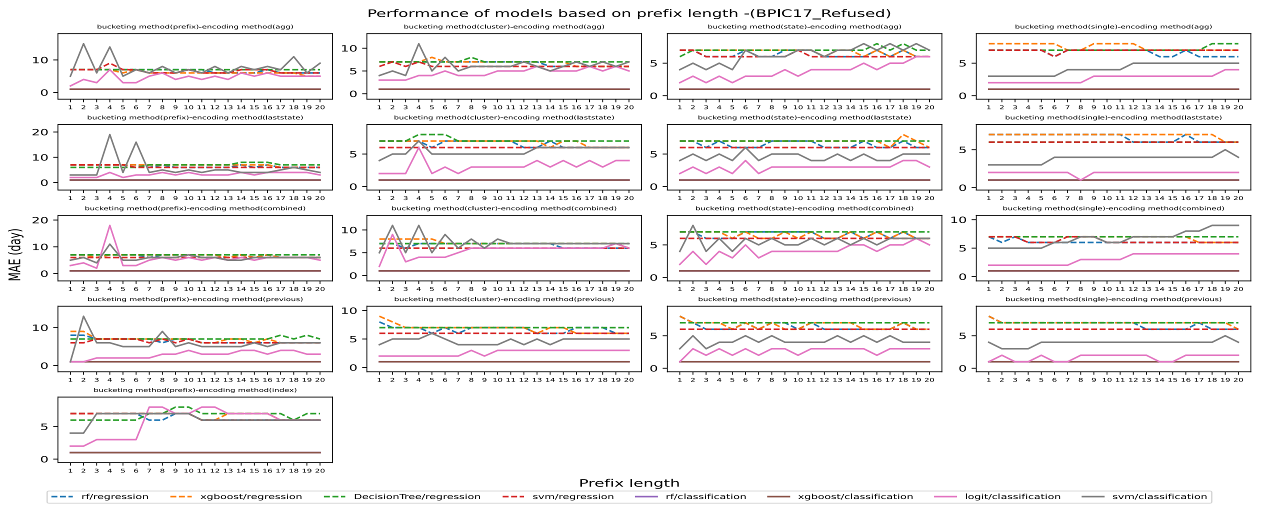

FIGURE 12. MAE of PPM models on BPIC17-Refused dataset by prefix length, categorized by bucketing and encoding methods.

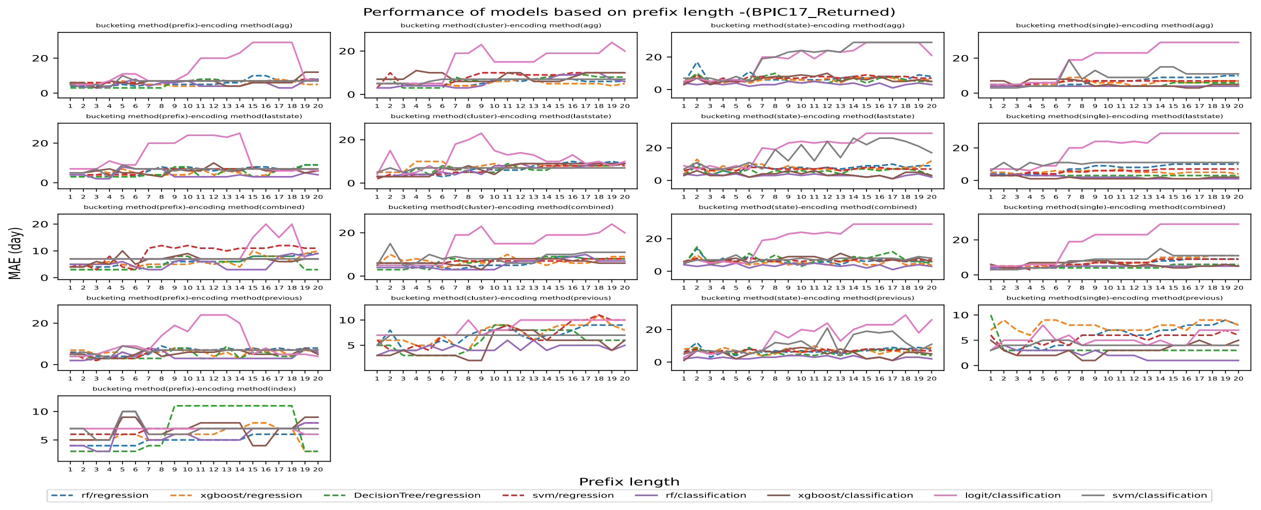

FIGURE 13. MAE of PPM models on BPIC17-Returned dataset by prefix length, categorized by bucketing and encoding methods.

algorithms as predictors. Therefore, models that are based on the CB-PPM method and employ RF classification algorithms are likely to be the most suitable models for time prediction.

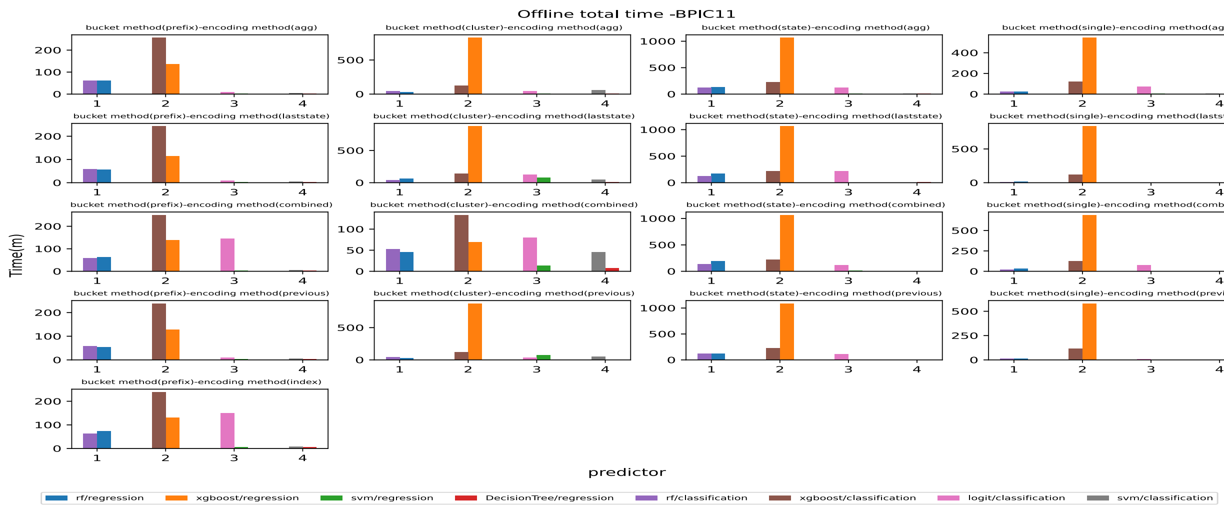

FIGURE 14. Offline total execution time of PPM models on BPIC11 dataset by prefix length, categorized by bucketing and encoding methods.

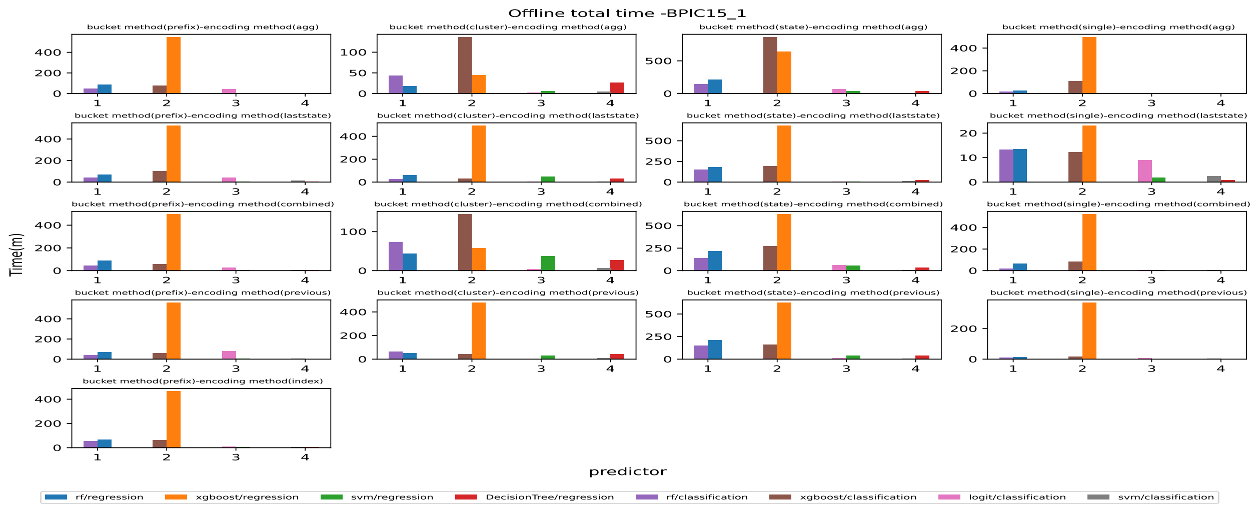

FIGURE 15. Offline total execution time of PPM models on BPIC15-1 dataset by prefix length, categorized by bucketing and encoding methods.

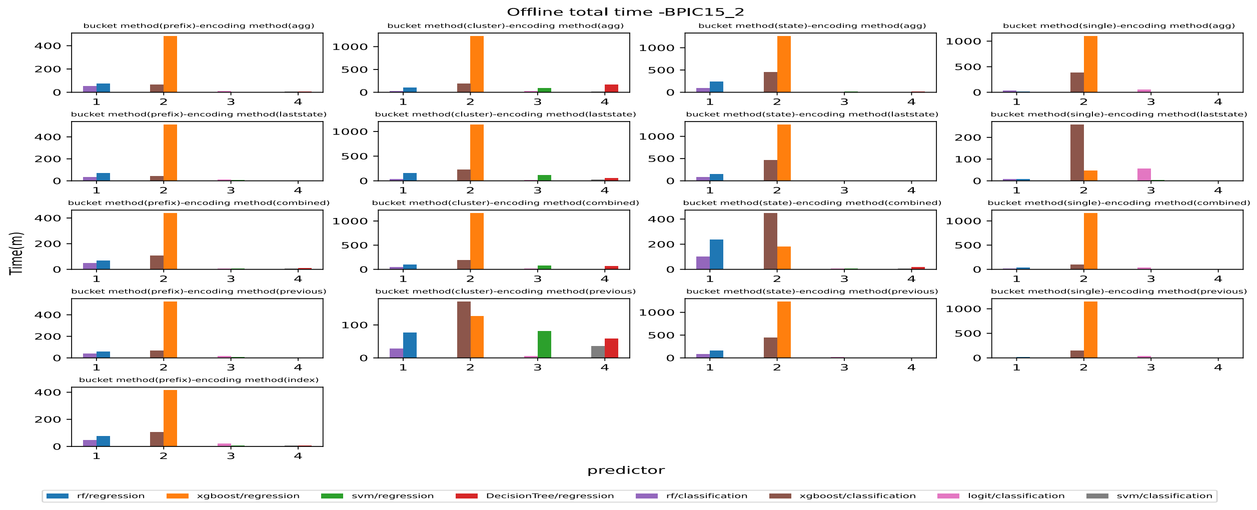

FIGURE 16. Offline total execution time of PPM models on BPIC15-2 dataset by prefix length, categorized by bucketing and encoding methods.

While this study provides useful insights into the effectiveness of different configurations of PPM models for time prediction in business processes, it has some limitations. For example, the use of only ten datasets in three

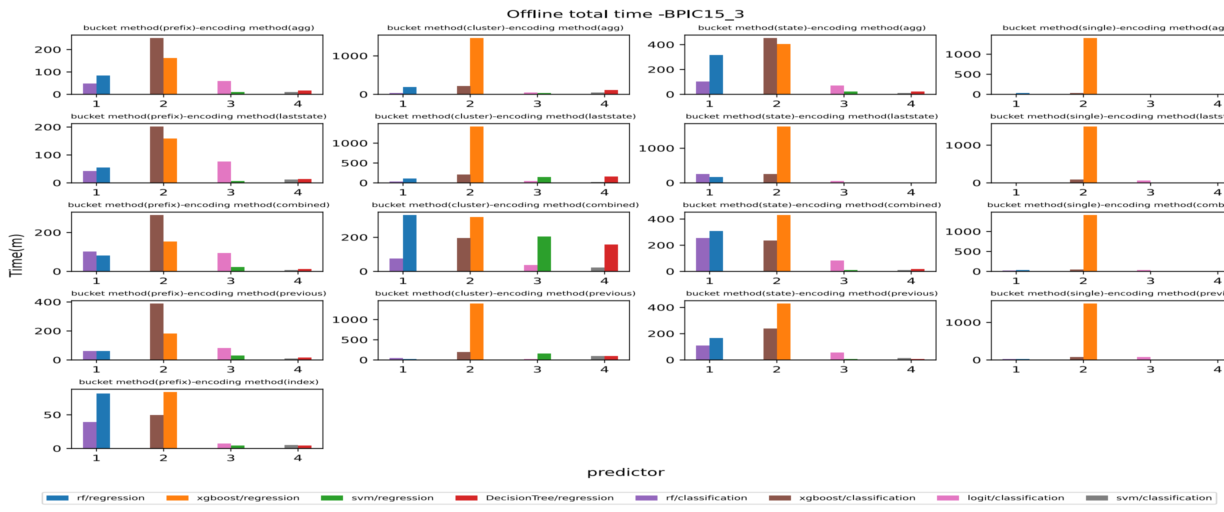

FIGURE 17. **Offline total execution time of PPM models on BPIC15-3 dataset by prefix length, categorized by bucketing and encoding methods.**

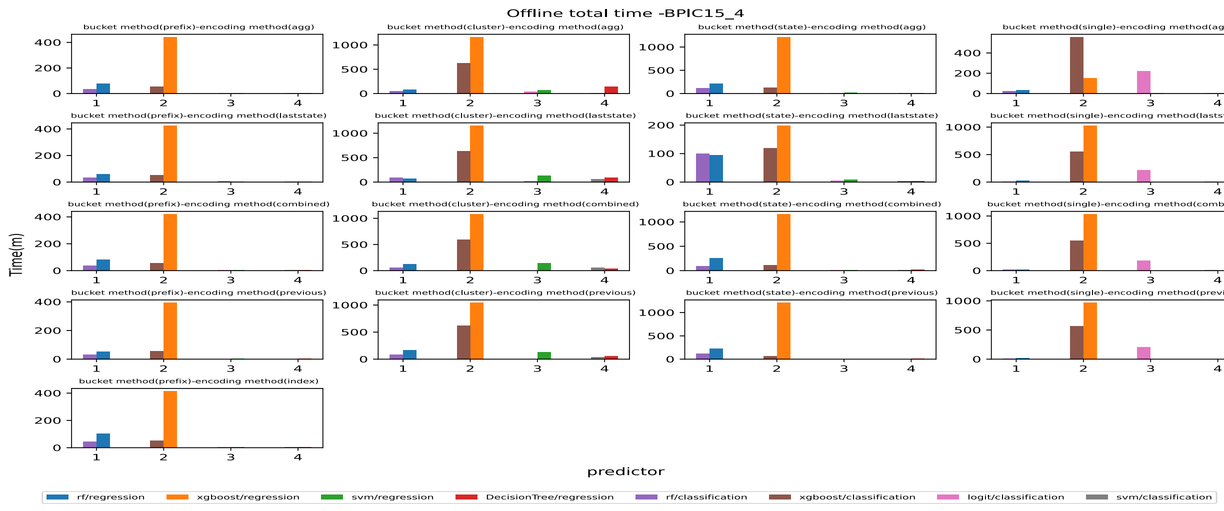

FIGURE 18. **Offline total execution time of PPM models on BPIC15-4 dataset by prefix length, categorized by bucketing and encoding methods.**

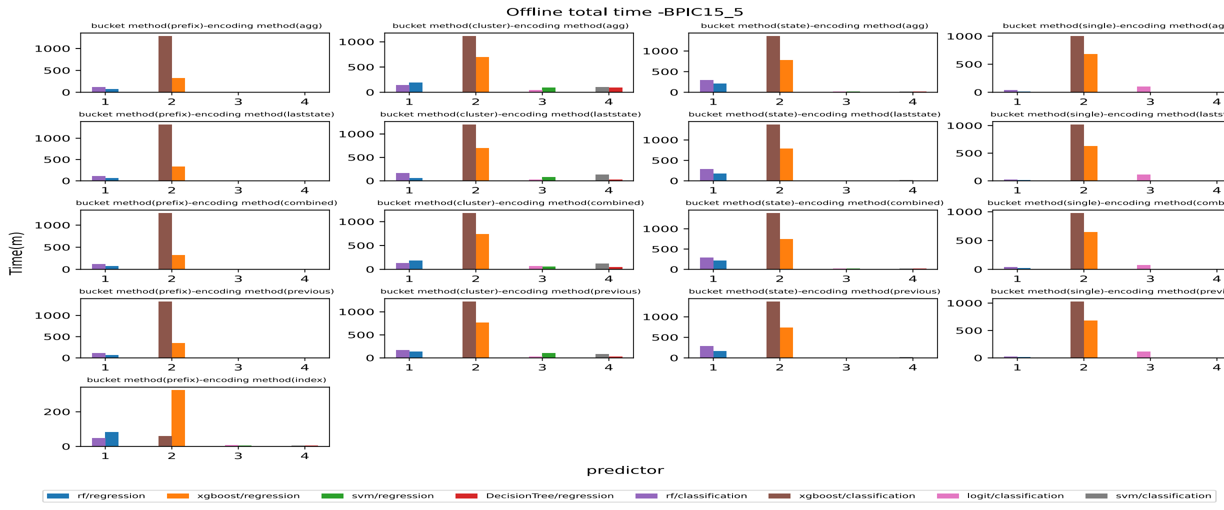

FIGURE 19. **Offline total execution time of PPM models on BPIC15-5 dataset by prefix length, categorized by bucketing and encoding methods.**

contexts limits the generalizability of the findings. Future research should explore the effect of process context on the accuracy and effectiveness of prediction methods, and consider more databases in various contexts to improve the

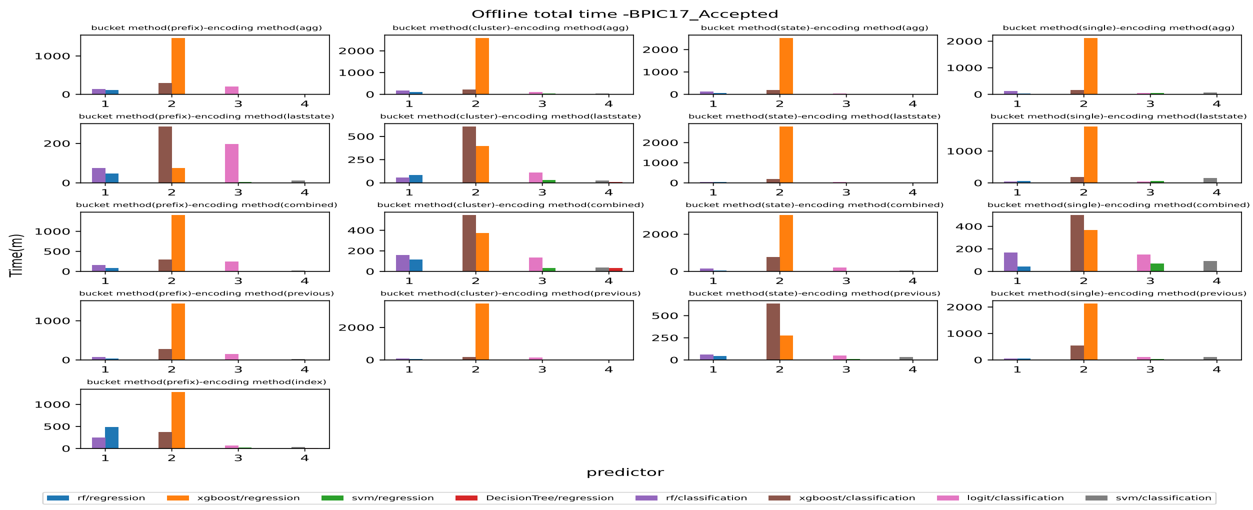

FIGURE 20. Offline total execution time of PPM models on BPIC17-Accepted dataset by prefix length, categorized by bucketing and encoding methods.

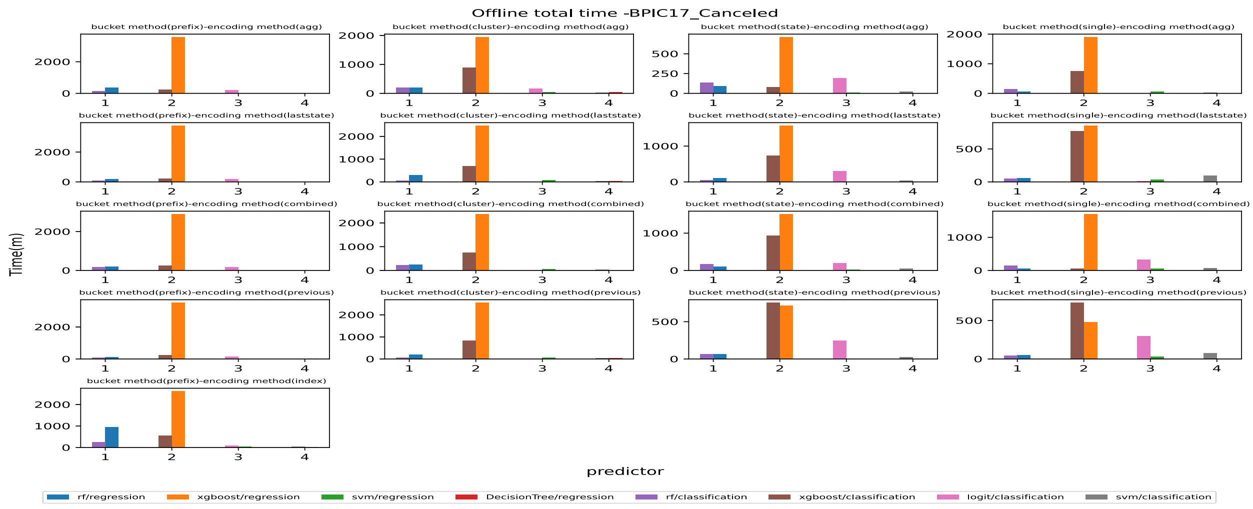

FIGURE 21. Offline total execution time of PPM models on BPIC17-Canceled dataset by prefix length, categorized by bucketing and encoding methods.

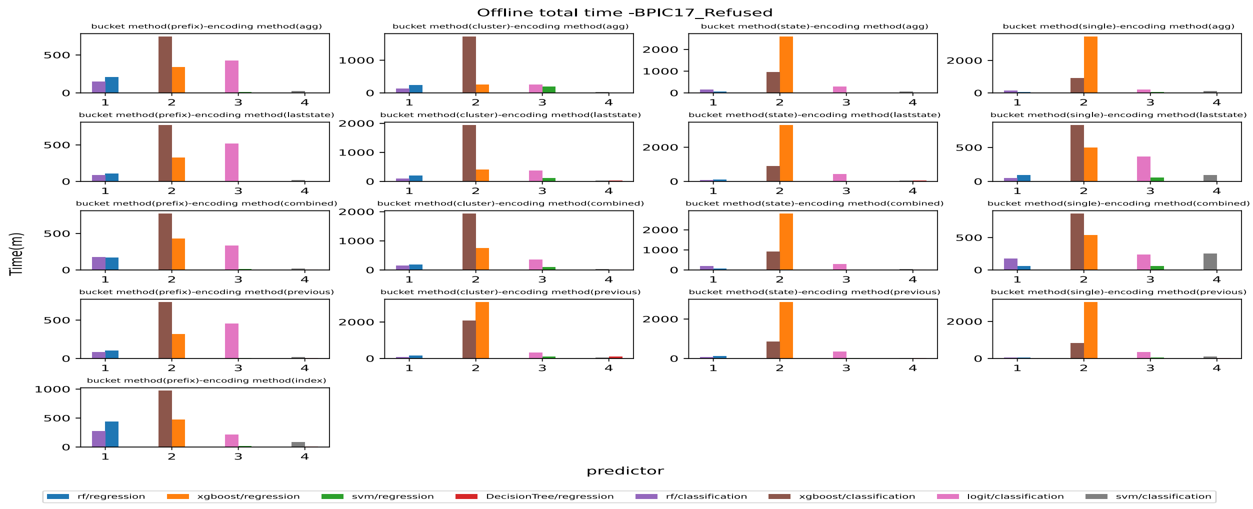

FIGURE 22. Offline total execution time of PPM models on BPIC17-Refused dataset by prefix length, categorized by bucketing and encoding methods.

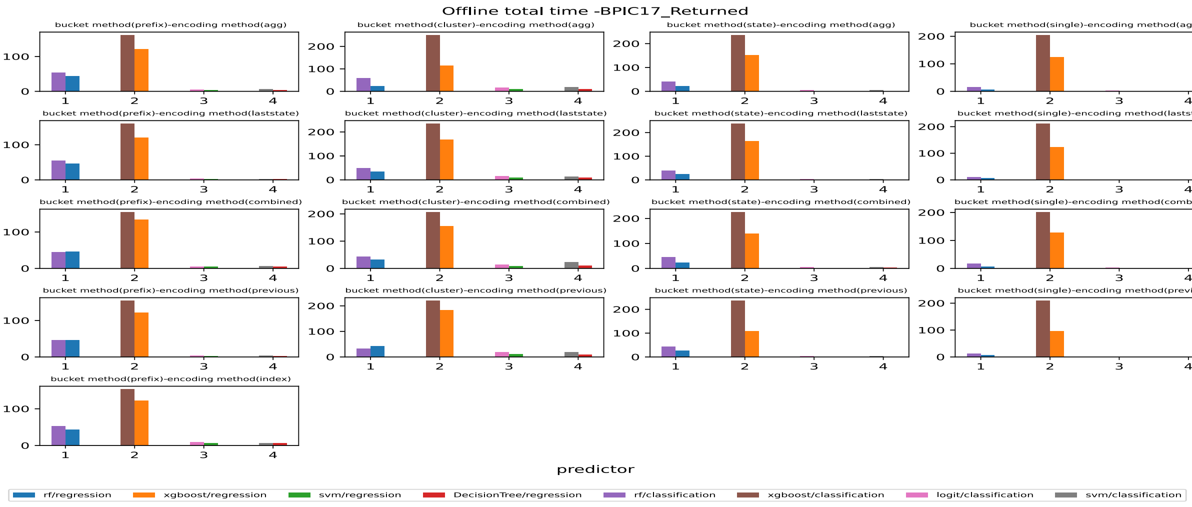

FIGURE 23. Offline total execution time of PPM models on BPIC17-Returned dataset by prefix length, categorized by bucketing and encoding methods.

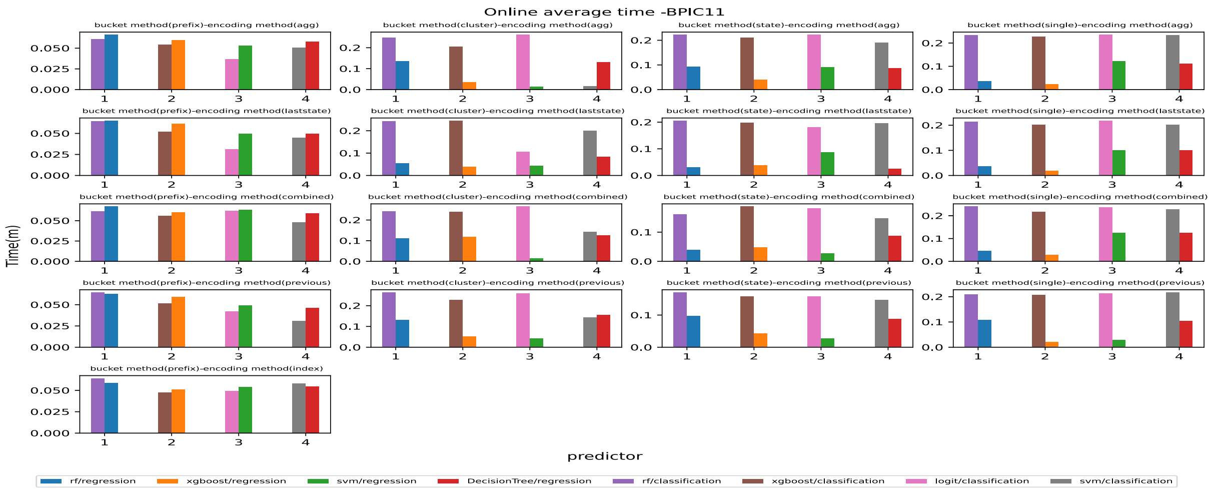

FIGURE 24. Online average execution time of PPM models on BPIC11 dataset by prefix length, categorized by bucketing and encoding methods.

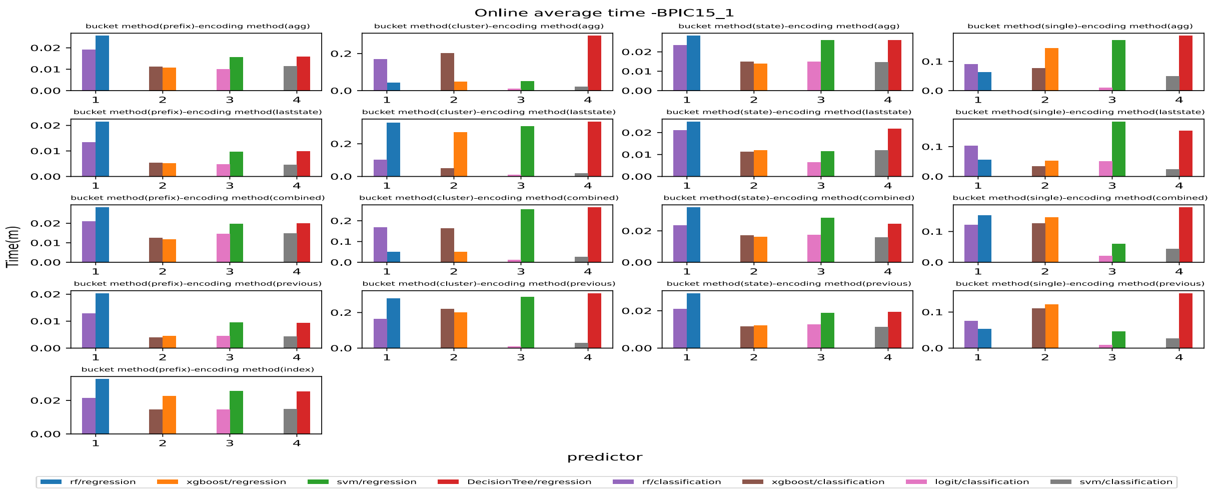

FIGURE 25. Online average execution time of PPM models on BPIC15-1 dataset by prefix length, categorized by bucketing and encoding methods.

comprehensiveness of results. To confirm the applicability of the CB-PPM and the RB-PPM methods, it is recommended to implement it in case studies and verify the results. Despite these limitations, the CB-PPM method exhibits promising

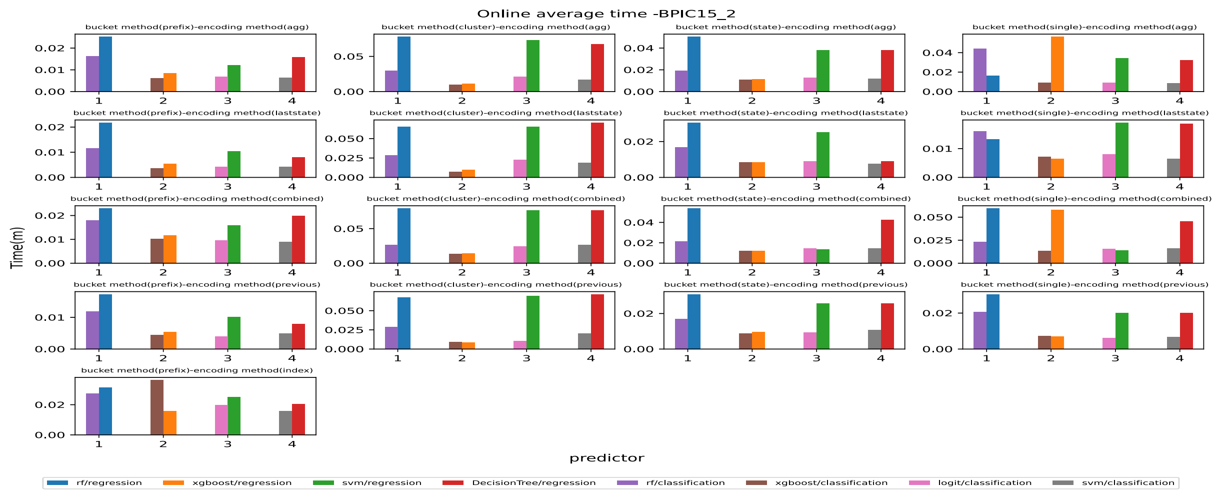

FIGURE 26. **Online average execution time of PPM models on BPIC15-2 dataset by prefix length, categorized by bucketing and encoding methods.**

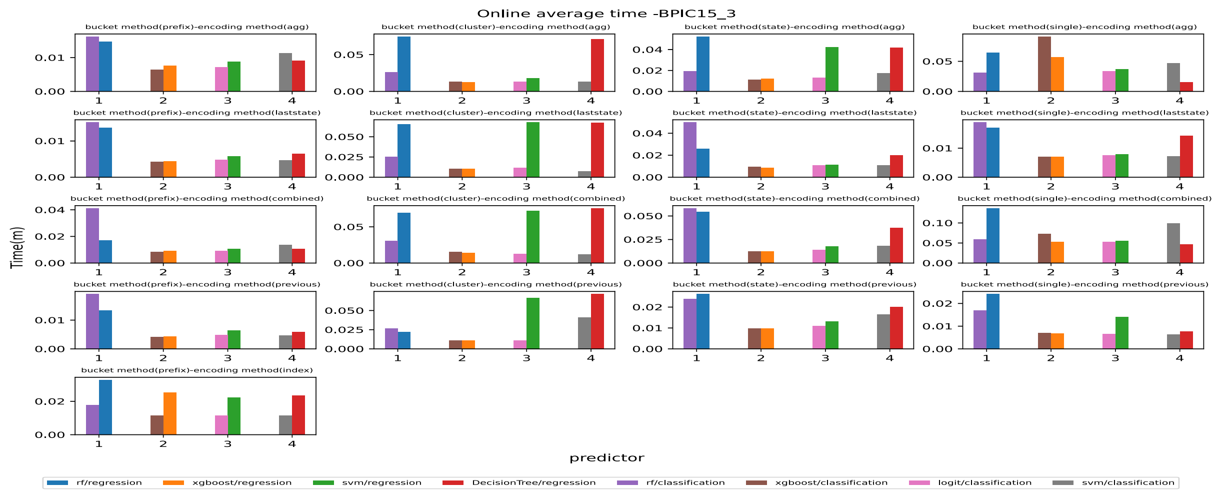

FIGURE 27. **Online average execution time of PPM models on BPIC15-3 dataset by prefix length, categorized by bucketing and encoding methods.**

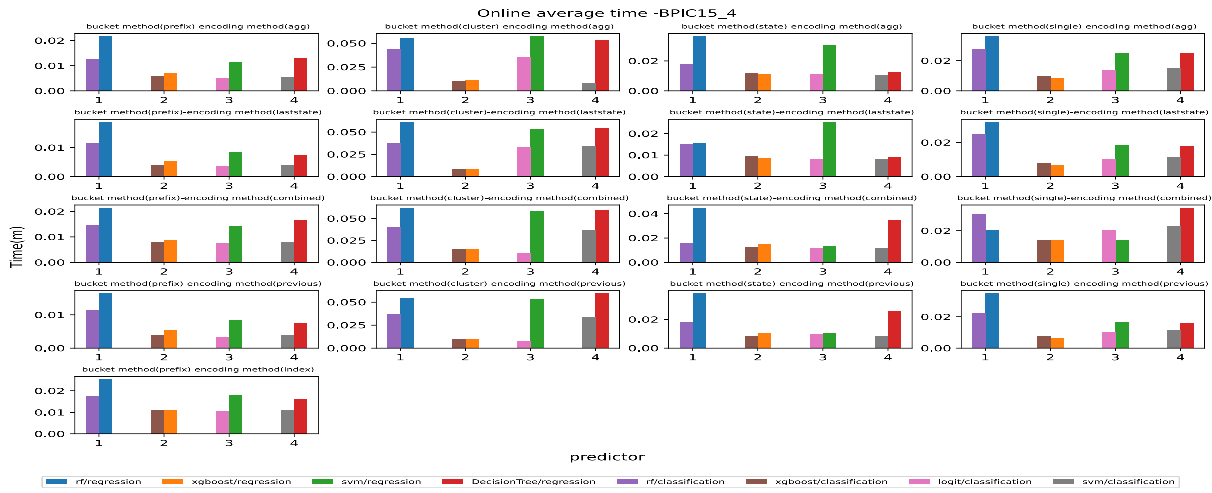

FIGURE 28. **Online average execution time of PPM models on BPIC15-4 dataset by prefix length, categorized by bucketing and encoding methods.**

potential to enhance time prediction accuracy compared to the RB-PPM method. This capability makes it a valuable tool for process managers and practitioners seeking to predict the remaining time of running cases and prevent deadline

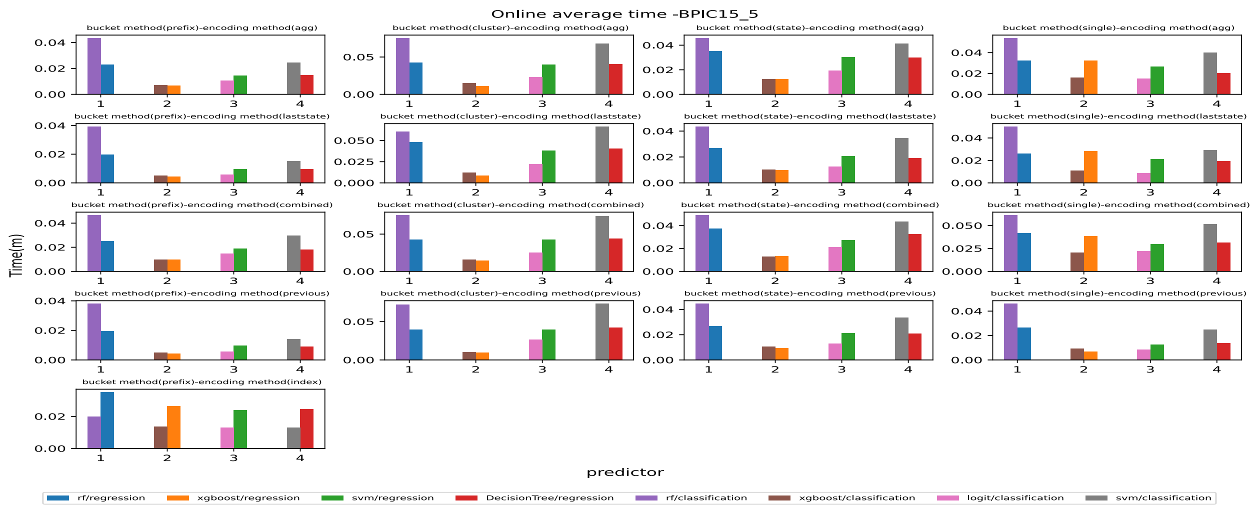

FIGURE 29. Online average execution time of PPM models on BPIC15-5 dataset by prefix length, categorized by bucketing and encoding methods.

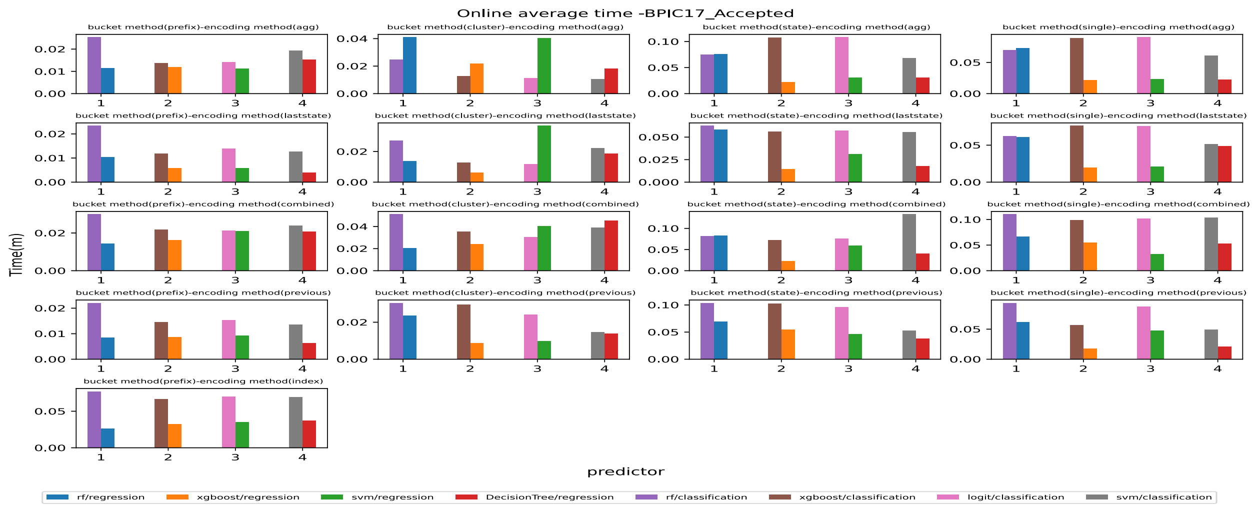

FIGURE 30. Online average execution time of PPM models on BPIC17-Accepted dataset by prefix length, categorized by bucketing and encoding methods.

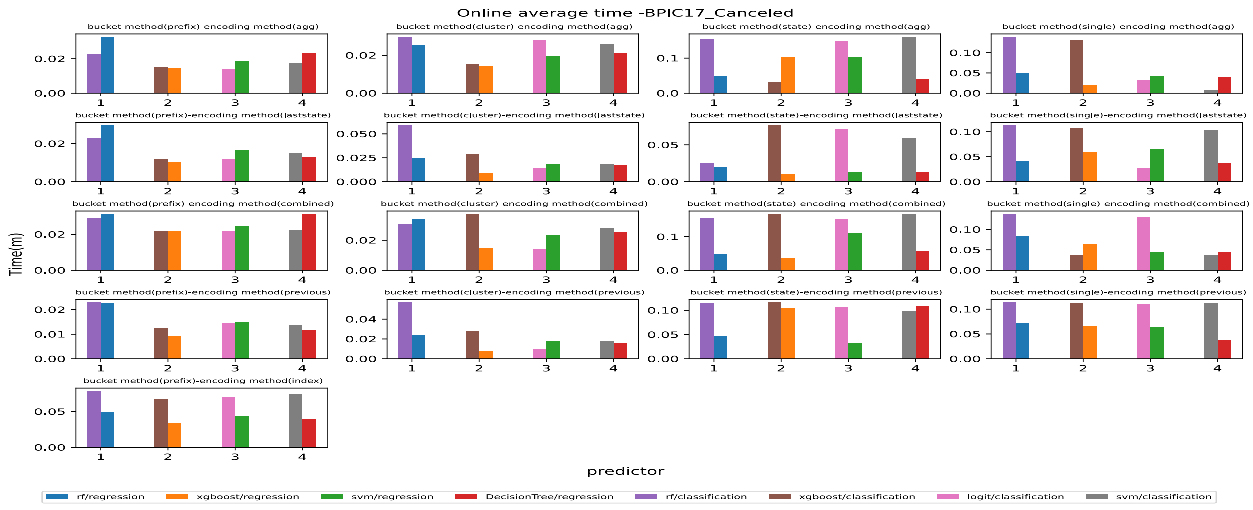

FIGURE 31. Online average execution time of PPM models on BPIC17-Canceled dataset by prefix length, categorized by bucketing and encoding methods.

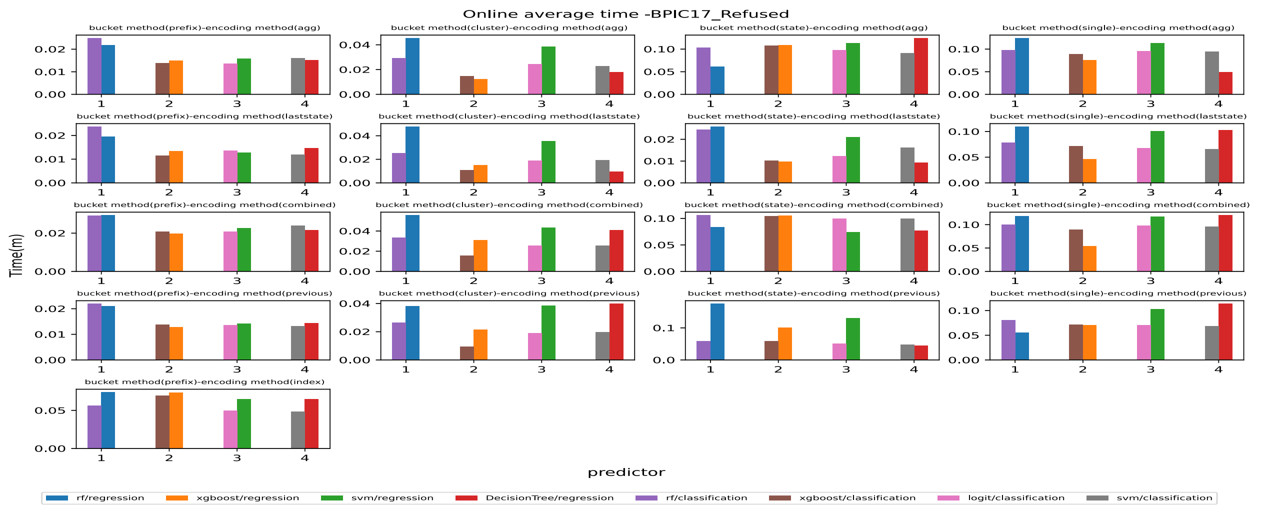

FIGURE 32. Online average execution time of PPM models on BPIC17-Refused dataset by prefix length, categorized by bucketing and encoding methods.

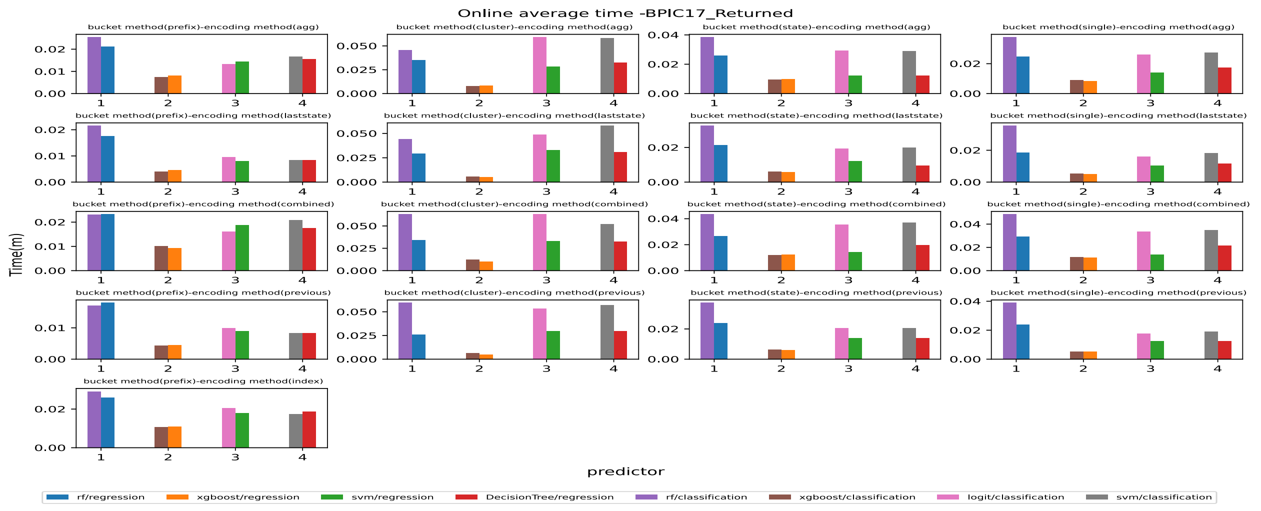

FIGURE 33. Online average execution time of PPM models on BPIC17-Returned dataset by prefix length, categorized by bucketing and encoding methods.

violations. Overall, the findings of this study can help organizations enhance the efficiency and effectiveness of their business processes.

VI. CONCLUSION

In conclusion, the accurate and time-efficient prediction of remaining time is crucial for effective business process management. The primary objective of our study was to apply the CB-PPM method for predicting remaining time and to conduct a comprehensive comparison of its advantages and disadvantages with the RB-PPM method. In addition, our study aimed to prioritize various PPM models by simultaneously considering MAE, offline execution time, and online execution time metrics through TOPSIS analysis. In our study, we configured and implemented a total of 136 distinct PPM models. These models were created by combining four bucketing methods, five encoding methods, and eight prediction algorithms, based on both CB-PPM and RB-PPM methods. These models were then evaluated across ten real-world datasets, providing comprehensive insights into the advantages and disadvantages of the CB-PPM and RB-PPM methods for time prediction.

Our hypothesis testing results have confirmed that the utilization of the CB-PPM method leads to improved prediction accuracy compared to the RB-PPM method. Furthermore, while the results of the hypothesis test indicated an increase in online execution time when the CB-PPM method is applied, there was no significant increase in offline execution time. TOPSIS analysis further supported the superiority of the CB-PPM method for time prediction, as 90% of the most suitable models were based on the CB-PPM method. Moreover, the results showed that the PPM model configuration strongly depends on the selected dataset.

While the study results confirmed the superiority of the CB-PPM method, further validation is needed to assess its generalizability in different contexts. Considering the preprocessing execution time in calculating offline execution time can also improve its accuracy. However, the careful selection of the appropriate PPM method represents a powerful tool for process managers and practitioners, enabling them to predict process remaining time accurately and optimize process scheduling effectively. The tangible benefits of appropriately configuring PPM models, such as cost savings, increased productivity, and improved customer satisfaction, highlight its potential to significantly enhance organizational performance. Overall, this study suggests that the CB-PPM method has the potential to improve business process management and drive success in various industries.

Our study contributes significantly to enhancing the accuracy and time-efficiency of remaining time prediction in business processes. This is achieved through the careful selection of the appropriate PPM approach and suitably configuring PPM models. Future research can contribute to a deeper understanding of the advantages of PPM approaches by exploring the potential of deep learning predictive process mining approach, to improve time prediction accuracy, offline execution time, and online execution time in business processes. Further research can explore the effect of process context on the performance of the CB-PPM and RB-PPM approaches. Moreover, conducting case studies to apply these approaches to time prediction could validate their practical applicability and effectiveness in real-world scenarios.

APPENDIX

In this section, we present details of the implemented models based on different criteria. It is important to note that the notation ‘Agg’ used for aggregation encoding, ‘Laststate’ for last state encoding, ‘Combined’ for combined encoding, ‘Previous’ for previous encoding, and ‘Index’ for index-based encoding. Additionally, ‘Classification’ denotes models based on the CB-PPM method, while ‘Regression’ denotes models based on the RB-PPM method. It’s worth noting that the results of different models are categorized based on the bucketing and encoding method. Specifically, each column of subplots represents the same bucketing method, while each row of subplots corresponds to the same encoding method.

APPENDIX A DETAILS OF MODELS’ MAE

The MAE of the implemented PPM models on different datasets based on prefix lengths is depicted in Figure 4 to 13.

APPENDIX B DETAILS OF MODELS’ OFFLINE TIME

The offline execution time of the implemented PPM models on various datasets is illustrated in Figures 14 to 23.

APPENDIX C DETAILS OF MODELS’ ONLINE TIME

The online execution time of implemented PPM models on different datasets is drawn in Figure 24 to 33.

REFERENCES

[1] W. Van Der Aalst, *Process Mining: Data Science in Action*, vol. 2. Cham, Switzerland: Springer, 2016.

[2] M. Dumas, M. La Rosa, J. Mendling, and H. A. Reijers, *Fundamentals of Business Process Management*, vol. 1. Cham, Switzerland: Springer, 2013.

[3] I. Verenich, M. Dumas, M. L. Rosa, F. M. Maggi, and I. Teinemaa, ‘‘Survey and cross-benchmark comparison of remaining time prediction methods in business process monitoring,’’ *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 4, pp. 1–34, Jul. 2019.

[4] I. Teinemaa, M. Dumas, M. L. Rosa, and F. M. Maggi, ‘‘Outcome-oriented predictive process monitoring: Review and benchmark,’’ *ACM Trans. Knowl. Discovery from Data*, vol. 13, no. 2, pp. 1–57, Apr. 2019.

[5] N. Ogunbiyi, A. Basukoski, and T. Chaussalet, ‘‘Investigating social contextual factors in remaining-time predictive process monitoring—A survival analysis approach,’’ *Algorithms*, vol. 13, no. 11, p. 267, Oct. 2020.

[6] M. Harl, S. Weinzierl, M. Stierle, and M. Matzner, ‘‘Explainable predictive business process monitoring using gated graph neural networks,’’ *J. Decis. Syst.*, vol. 29, no. sup1, pp. 312–327, Aug. 2020.

[7] X. Sun, W. Hou, Y. Ying, and D. Yu, ‘‘Remaining time prediction of business Processes based on multilayer machine learning,’’ in *Proc. IEEE Int. Conf. Web Services (ICWS)*, Oct. 2020, pp. 554–558.

[8] A. Aburomman, M. Lama, and A. Bugarín, ‘‘A vector-based classification approach for remaining time prediction in business processes,’’ *IEEE Access*, vol. 7, pp. 128198–128212, 2019.

[9] A. C. Choueiri, D. M. V. Sato, E. E. Scalabrin, and E. A. P. Santos, ‘‘An extended model for remaining time prediction in manufacturing systems using process mining,’’ *J. Manuf. Syst.*, vol. 56, pp. 188–201, Jul. 2020.

[10] I. Teinemaa, M. Dumas, A. Leontjeva, and F. M. Maggi, ‘‘Temporal stability in predictive process monitoring,’’ *Data Mining Knowl. Discovery*, vol. 32, no. 5, pp. 1306–1338, Sep. 2018.

[11] A. E. Márquez-Chamorro, M. Resinas, and A. Ruiz-Cortés, ‘‘Predictive monitoring of business processes: A survey,’’ *IEEE Trans. Services Comput.*, vol. 11, no. 6, pp. 962–977, Nov. 2018.

[12] C. Di Francescomarino, C. Ghidini, F. M. Maggi, and F. Milani, ‘‘Predictive process monitoring methods: Which one suits me best?’’ in *Proc. Int. Conf. Bus. Process Manage.* Sydney, NSW, Australia: Springer, 2018, pp. 462–479.

[13] A. Dogan and D. Birant, ‘‘Machine learning and data mining in manufacturing,’’ *Exp. Syst. Appl.*, vol. 166, Mar. 2021, Art. no. 114060.

[14] C. Di Francescomarino, M. Dumas, M. Federici, C. Ghidini, F. M. Maggi, W. Rizzi, and L. Simonetto, ‘‘Genetic algorithms for hyperparameter optimization in predictive business process monitoring,’’ *Inf. Syst.*, vol. 74, pp. 67–83, May 2018.

[15] N. Tax, I. Verenich, M. La Rosa, and M. Dumas, ‘‘Predictive business process monitoring with LSTM neural networks,’’ in *Advanced Information Systems Engineering*, E. Dubois and K. Pohl, Eds. Cham, Switzerland: Springer, 2017, pp. 477–492.

[16] A. Cuzzocrea, F. Folino, M. Guarascio, and L. Pontieri, ‘‘Predictive monitoring of temporally-aggregated performance indicators of business processes against low-level streaming events,’’ *Inf. Syst.*, vol. 81, pp. 236–266, Mar. 2019.

[17] J. Kim and M. Comuzzi, ‘‘A diagnostic framework for imbalanced classification in business process predictive monitoring,’’ *Expert Syst. Appl.*, vol. 184, Dec. 2021, Art. no. 115536.

[18] C. D. S. Garcia, A. Meincheim, E. R. Faria Junior, M. R. Dallagassa, D. M. V. Sato, D. R. Carvalho, E. A. P. Santos, and E. E. Scalabrin, ‘‘Process mining techniques and applications—A systematic mapping study,’’ *Exp. Syst. Appl.*, vol. 133, pp. 260–295, Nov. 2019.

[19] M. Camargo, D. Báron, M. Dumas, and O. González-Rojas, ‘‘Learning business process simulation models: A hybrid process mining and deep learning approach,’’ *Inf. Syst.*, vol. 117, Jul. 2023, Art. no. 102248.

[20] M. Polato, A. Sperduti, A. Burattin, and M. D. Leoni, ‘‘Time and activity sequence prediction of business process instances,’’ *Computing*, vol. 100, no. 9, pp. 1005–1031, Sep. 2018.

[21] F. Folino, M. Guarascio, and L. Pontieri, ‘‘Discovering context-aware models for predicting business process performances,’’ in *Proc. OTM Confederated Int. Conf. Meaningful Internet Syst.* Cham, Switzerland: Springer, 2012, pp. 287–304.

[22] I. Firouzian, M. Zahedi, and H. Hassanpour, ‘‘Investigation of the effect of concept drift on data-aware remaining time prediction of business processes,’’ *Int. J. Nonlinear Anal. Appl.*, vol. 10, no. 2, pp. 153–166, 2019.

[23] W. M. P. van der Aalst, M. H. Schonenberg, and M. Song, ‘‘Time prediction based on process mining,’’ *Inf. Syst.*, vol. 36, no. 2, pp. 450–475, Apr. 2011.

[24] A. Rogge-Solti and M. Weske, "Prediction of remaining service execution time using stochastic Petri nets with arbitrary firing delays," in *Proc. Service-Oriented Comput., 11th Int. Conf.*, Berlin, Germany. Cham, Switzerland: Springer, 2013, pp. 389–403.
[25] M. Polato, A. Sperduti, A. Burattin, and M. de Leoni, "Data-aware remaining time prediction of business process instances," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2014, pp. 816–823.
[26] M. Ceci, P. F. Lanotte, F. Fumarola, D. P. Cavallo, and D. Malerba, "Completion time and next activity prediction of processes using sequential pattern mining," in *Proc. 17th Int. Conf.*, Bled, Slovenia. Cham, Switzerland: Springer, Oct. 2014, pp. 49–61.
[27] H. Weytjens and J. De Weerdt, "Learning uncertainty with artificial neural networks for predictive process monitoring," *Appl. Soft Comput.*, vol. 125, Aug. 2022, Art. no. 109134.
[28] N. A. Wahid, T. N. Adi, H. Bae, and Y. Choi, "Predictive business process monitoring-remaining time prediction using deep neural network with entity embedding," *Proc. Comput. Sci.*, vol. 161, pp. 1080–1088, 2019.
[29] A. Senderovich, C. D. Francescomarino, C. Ghidini, K. Jorbina, and F. M. Maggi, "Intra and inter-case features in predictive process monitoring: A tale of two dimensions," in *Proc. Int. Conf. Bus. Process Manage.* Barcelona, Spain: Springer, 2017, pp. 306–323.
[30] N. A. Wahid, H. Bae, T. N. Adi, Y. Choi, and Y. A. Iskandar, "Parallel-structure deep learning for prediction of remaining time of process instances," *Appl. Sci.*, vol. 11, no. 21, p. 9848, Oct. 2021.
[31] A. E. Márquez-Chamorro, M. Resinas, A. Ruiz-Cortés, and M. Toro, "Run-time prediction of business process indicators using evolutionary decision rules," *Exp. Syst. Appl.*, vol. 87, pp. 1–14, Nov. 2017.
[32] B. R. Gunnarsson, S. V. Broucke, and J. D. Weerdt, "A direct data aware LSTM neural network architecture for complete remaining trace and runtime prediction," *IEEE Trans. Services Comput.*, vol. 16, no. 4, pp. 1–13, 2023.
[33] J. Han, M. Kamber, and J. Pei, *Data Mining Concepts and Techniques*, 3rd ed. San Francisco, CA, USA: Morgan Kaufmann, 2012.
[34] C. D. Francescomarino, M. Dumas, F. M. Maggi, and I. Teinemaa, "Clustering-based predictive process monitoring," *IEEE Trans. Services Comput.*, vol. 12, no. 6, pp. 896–909, Nov. 2019.
[35] D. C. Montgomery, *Design and Analysis of Experiments*, 9th ed. Hoboken, NJ, USA: Wiley, 2017.
[36] A. Revina and Ü. Aksu, "An approach for analyzing business process execution complexity based on textual data and event log," *Inf. Syst.*, vol. 114, Mar. 2023, Art. no. 102184.
[37] F. Faul, E. Erdfelder, A.-G. Lang, and A. Buchner, "G*Power 3: A flexible statistical power analysis program for the social, behavioral, and biomedical sciences," *Behav. Res. Methods*, vol. 39, no. 2, pp. 175–191, May 2007.
[38] J. Kim, M. Comuzzi, M. Dumas, F. M. Maggi, and I. Teinemaa, "Encoding resource experience for predictive process monitoring," *Decis. Support Syst.*, vol. 153, Feb. 2022, Art. no. 113669.
[39] D. Impedovo, G. Pirlo, and G. Semeraro, "Next activity prediction: An application of shallow learning techniques against deep learning over the BPI challenge 2020," *IEEE Access*, vol. 11, pp. 117947–117953, 2023.
[40] M. Behzadian, S. Khanmohammadi Otaghsara, M. Yazdani, and J. Ignatius, "A state-of the-art survey of TOPSIS applications," *Exp. Syst. Appl.*, vol. 39, no. 17, pp. 13051–13069, Dec. 2012.

REZA AALIKHANI received the Bachelor of Mathematics degree from Arak University, Arak, Iran, in 2015, and the Master of Industrial Engineering degree in economic-social systems from Iran University of Science and Technology, Tehran, in 2018, where he is currently pursuing the Ph.D. degree, applying his expertise in intelligent service composition and predictive process monitoring to the realm of healthcare. His research interests include optimizing patient care workflows, predicting and optimizing patient outcomes, and improving resource management and care coordination within healthcare systems.

MOHAMMAD FATHIAN received the M.S. and Ph.D. degrees in industrial engineering from Iran University of Science and Technology, Tehran. He is currently a Professor with the School of Industrial Engineering, Iran University of Science and Technology. He is working in the areas of information technology and industrial engineering. He has more than 90 journal articles and five books in the areas of industrial engineering and information technology.

MOHAMMAD REZA RASOULI received the Bachelor of Industrial Engineering degree from the Amirkabir University of Technology, Tehran, Iran, in 2005, the master's degree in industrial engineering from Shahed University, Tehran, in 2008, and the Ph.D. degree in industrial engineering from Eindhoven University of Technology, The Netherlands, in 2015. He is currently an Assistant Professor with Iran University of Science and Technology, Tehran, where he is also the Director of the System Engineering Department, Industrial Engineering Faculty. His research interests include information technology and industrial engineering.