Assignment 2 for APML2022

Group 9
Christian Acosta | Josh Bleijenberg | Mischa van Ek | Sjoerd Vink | Robin Wiersma

Abstract

This paper studies different algorithms for predictive process monitoring. The algorithms used are decision tree regression, kNN regression and random forest regression. The algorithms are applied to a data set provided by a hospital. The goal was to optimize the process flow by being able to predict the remaining time someone has to stay in the hospital. The first step was preprocessing the data which is done in two ways, namely last state encoding and aggregation encoding. The performance of each respective model and encoding combination was tested using various performance measures. Performance results showed that the regression tree, in combination with aggregated-state encoding, delivered the best performance for this data set.

1 Introduction

Just like any other data-driven company, a hospital stores large amounts of data. Data that is not yet being used to improve internal hospital processes. This is done, for example, when predicting the remaining time of a patient in the hospital. This way, a doctor can choose medical procedures more carefully. In this paper, previous medical actions are looked at to predict the remaining time someone has to stay in the hospital. This gives a hospital new insights and can therefore make better decisions. These predictions are made using regression models and are analyzed based on of various metrics. To get an overview of the process, a Directly-Follows Graph and a BPMN model have been made. These are visible in Appendix 1.

2 Data

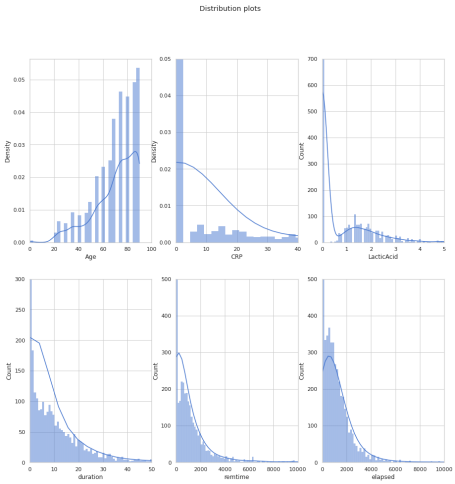
The provided data set is a real-life event log containing events of sepsis cases from a hospital. Sepsis is a life threatening condition, typically caused by an infection. Every case represents a patient's pathway through the treatment process. The events were recorded by the ERP (Enterprise Resource Planning) system of the hospital. The original data set contains about 1000 cases with in total 5.000 events that were recorded for 16 different activities. Moreover, 39 data attributes are recorded, e.g., the group responsible for the activity, the results of tests and information from checklists. Events and attribute values have been anonymized. The time stamps of events have been randomized, but the time between events within a trace has not been altered.

All variables are presented in Table 1. The input variables consist of triage documents filled in for the patients with information on the time the triage was conducted (disfunctional organ, hypotension, hypoxia, suspected infection, oliguria), the symptoms present (six SIRS criteria), the ordered diagnostics (eleven diagnostics) and the time infusions and antibiotics were administered (intravenous infusion). The laboratory also supplied blood tests as input data (leucocytes measurement, CRP measurement, lactic-acid measurement). The target data is the further trajectory of the patient.

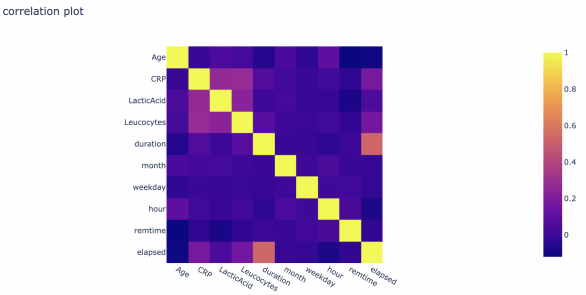
Looking at the distribution of several dimensions in Figure 1a it can be seen that the plots are somewhat normally distributed. It is true that these distribution plots are skewed. With age, for example, it can be seen that the frequency increases as the patients get older. This may be due to

Preprint. Under review.

the fact that sepsis occurs more often in older people than in young people. Figure 1b shows the correlations between the different dimensions in the data set. The darker the area, the lower the correlation. It is remarkable that no dimensions are strongly correlated with each other.



(a) Data distribution



(b) Correlation plot

Figure 1: Data analysis

Variable	Data type	Variable	Data type
DiagnosticArtAstrup	object	SIRSCritLeucos	object
DiagnosticBlood	object	SIRSCritTachypnea	object
DiagnosticECG	object	SIRSCritTemperature	object
DiagnosticIC	object	SIRSCriteria2OrMore	object
DiagnosticLacticAcid	object	Age	float64
DiagnosticLiquor	object	Case ID	object
DiagnosticOther	object	Activity	object
DiagnosticSputum	object	Diagnose	object
DiagnosticUrinaryCulture	object	org:group	object
DiagnosticUrinarySediment	object	CRP	float64
DiagnosticXthorax	object	LacticAcid	float64
DisfuncOrg	object	Leucocytes	float64
Hypotensie	object	Complete Timestamp	object
Hypoxie	object	duration	float64
InfectionSuspected	object	month	int64
Infusion	object	weekday	int64
Oligurie	object	hour	int64
SIRSCritHeartRate	object	remtime	float64
SIRSCritHeartRate	object	elapsed	float64

Table 1: Overview of variables and data types

3 Methods

Several regression algorithms have been implemented for predictive process monitoring tasks on the given data set. All algorithms are derived from the Scikit-learn library (version 1.0.1). For evaluation purposes, the data is split into training and testing data with a test-size of 0.33. The model is trained with the training data and the evaluation metrics are calculated with the test data.

Figure 2: Preprocessing pipeline

3.1 Preprocessing steps

In order for the algorithms to work as well as possible, it is important that the data is well prepared. To do this, several steps have been taken. An overview of the taken steps is displayed in Figure 2

Data cleaning The first step is to remove the NaN values. Because the domain is unknown, no statement can be made about the most likely values with which it can be filled. For this reason, it was decided to drop the rows containing NaN values. The data set is large enough to do this. So there is enough left over.

Label encoding In order to be able to perform computations on the data, it is important that no textual (categorical) values can be found in the data set. The original data set contains such values, which makes it important to do something about it. A label encoder is used to transform these categorical values into numerical values.

Normalization Because different regression algorithms are used later in the document, it is important to normalize the data set. Although it does not matter for a decision tree whether the input data is normalized, it does for a kNN regression. For this reason, a normalization has been applied. By applying a normalization, all values in the data set are brought between 0 and 1.

3.2 Trace encoding

Because it is a predictive process monitoring task, it is necessary to apply some form of trace encoding. In this way, not only a single row is taken into account, but also the activity that happened before it. A distinction can be made between two types of trace encoding that are both applied. Ultimately, it can be reported which produces better results. Both methods are explained in this section.

Last state encoding Last state encoding is a principle that, for each data point, takes into account the previous activity in a sequence. In this way, data points are not viewed individually, but together with the previous activity. Intuitively, the essential idea of last state encoding is that only the last event is important and indicates the state of the case, previous events are less important. After this last state encoding, both the current and the previous state are hot encoded.

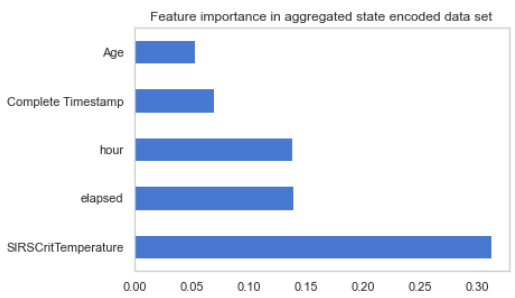
Aggregation state encoding Aggregation encoding is a principle that considers not only the previous state, but all states leading to the data point. The essential idea of aggregation coding is that only all past events can matter and affect the future state of affairs. In this way, a better result may be obtained. This principle is applied per category in the column activity.

3.3 Algorithms

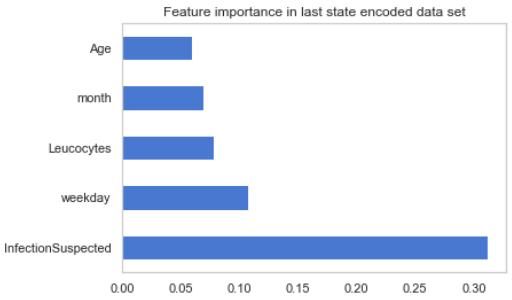
Regression is a predictive problem setting where historical data is used to predict continuous values. In order to be able to predict the remaining time in this study, various algorithms have been selected.

3.3.1 Decision tree regression

Intuitively, a regression tree is a decision tree used for regression. A regression tree can thus be used to predict continuous values. In the case of this research, it is about the remaining time. The regression tree tries to make nodes as small as possible by squared error in the leaf. For the actual prediction, the average of the leaf node that the regression tree leads to is taken. Care should be taken with overfitting. A regression tree with default parameters quickly grows very large. This not only makes it very slow, the test accuracy also becomes low because the model cannot be generalized. In a decision tree, each feature has a different importance. The features that have the greatest influence on the time remaining prediction are shown in Figure 3a and Figure 3b



(a) Feature importance - aggregated state encoding



(b) Feature importance - last-2-state encoding

3.3.2 K-nearest-neighbour regression

Where the kNN algorithm for classification is used, the kNN regression algorithm for regression can be used. The algorithms are fundamentally very similar. In kNN regression, the principle of most votes is not used, but the average of the specified number of neighbours. The kNN regression algorithm does not prepare a specified model in advance, unlike decision tree regression. In a kNN regression, the computation is postponed until the query is actually created. For this reason, the training times are very short. However, a trade-off must be made between variance and bias. Raising the k results in a lower variance but a higher bias, and vice versa.

3.3.3 Random forest regression

Finally, a random forest regression model is used to predict the remaining time. A random forest is very similar to a decision tree. The only difference is that various decision trees are trained in a random forest. In a regression task, the predictions of the decision trees are then averaged. This is a very powerful model, however, training times can sometimes be very long.

4 Overview of model performance

Following the performance measures, we can deduce that the optimal model for this data set is the decision tree regression model. This model, in combination with aggregated-state encoding, returned the lowest MAE score out of all model and encoding combinations.

The MAE was chosen as the performance measure for finding the optimal model. MAE is easier to interpret than (R)MSE, as MAE only describes the average error and has no further implications that make it difficult to understand.

Aggregated state encoding returned the lowest (best) MAE scores for both models. The difference between the encoding methods is quite noticeable, especially so for the regression tree model on the held out data (1121.39 versus 879.09).

As can be seen in figure 3a and figure 3b, the feature with highest importance is different for each encoding method, but they both have roughly the same amount of feature importance respectively.

In table 2 we present an overview of the performance of each model on the held out test data. For each model, two types of encoding have been tested, namely last-2-state and aggregated-state encoding.

The performance measures used are Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE) and R-squared (R^2).

Model	Encoding	Test MAE	Test MSE	Test RMSE	Test R^2
Regression Tree	Last-2-state	1121.39	7701301.62	2775.12	0.33
Regression Tree	Agg-state	879.09	9786716.9	3128.37	0.15
kNN	Last-2-state	1071.27	8049227.23	2837.12	0.30
kNN	Agg-state	939.77	7206421.94	2684.48	0.37

Table 2: Overview of model performance on held out test data using

In table 3 we present an overview of the cross-validated performance of each model using the validation data. The performance measures used are the same as in table 2

Model	Encoding	CV MAE	CV MSE	CV RMSE	CV R^2
Regression Tree	Last-2-state	1140.53	8415383.00	2900.93	0.27
Regression Tree	Agg-state	969.29	8139702.00	2853.02	0.32
kNN	Last-2-state	1096.74	8566563.00	2926.87	0.27
kNN	Agg-state	937.06	7430140.00	2725.83	0.37

Table 3: Overview of cross-validated model performance on validation data

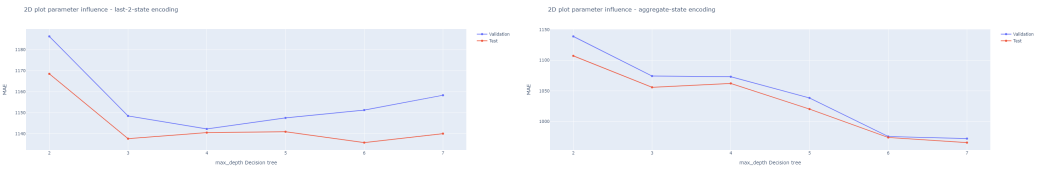
5 Evaluation and Discussion

In this chapter the experimental setup is described which is used to construct the optimal tree. Also, the results are described and discussed. For evaluation, the mean square error was used to decide which parameter setting offers the best performance on the given data set. The parameters who had the lowest mean square error were deemed as the most optimal parameters. In the table underneath, the functions required to calculate the Evaluation metrics: MSE, RMSE and MAE are shown.

Absolute error	$e = (\Delta x) = \lvert x_i - x \rvert$
Mean squared error	$\text{MSE} = \frac{1}{n}\sum_{t=1}^{n} e_t^2$
Root mean squared error	$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{t=1}^{n} e_t^2}$
Mean absolute error	$\text{MAE} = \frac{1}{n}\sum_{t=1}^{n} \lvert e_t \rvert$

5.1 Decision tree regression

For the decision tree regression, both preprocessing techniques, last-2-state and aggregated-state coding, were evaluated. The evaluation assessed two parameters. The first evaluation assessed how both the validation subset and the test subset responded to adjusting the maximum depth of a range from 2 to 8. These tests showed that the validation subgroup had a higher mean squared error than the test subgroup for both aggregated and last-2-state coding. Interestingly, the last-2-state encoding graph seen in figure 4a Shows the validation to increase the mean squared error, whereas test subset moderately stays the same. The rise of the mean square error in the validation subset is suspected to be caused by overfitting. In general, overfitting means that the model is too specific for the data set used to learn the model. Overfitting is caused by the increase of the maximum depth in the decision tree, which causes the classification function to be overly specific to a limited set of data points.



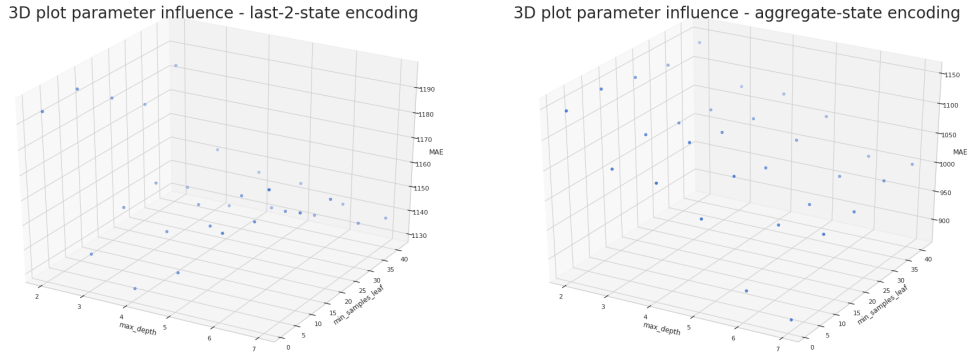
(a) 2D last-2-state encoding

(b) 2D aggregated encoding

Further analysis was done by also adjusting the minimal samples leaf. These tests revealed the optimal parameters based on mean absolute error are:

- Minimal samples leaf = 1
- Maximum depth = 4

A 10-fold cross validation was used for this to generalize the result to counter overfitting. The adjustments of the two parameters were both tested on the aggregated and last-2-state encoding datasets. Broadly speaking, we saw that maximum depth has more influence on the mean absolute error than minimal samples leaf. The most intriguing correlation seen in 3d plot displaying the parameter influence, is that last-2-state encoding shows to have a higher effect in figure 5b than aggregated encoding shown in figure 5b. The reasons for this result is not fully understood and would require further research to understand the root cause of this deviation.



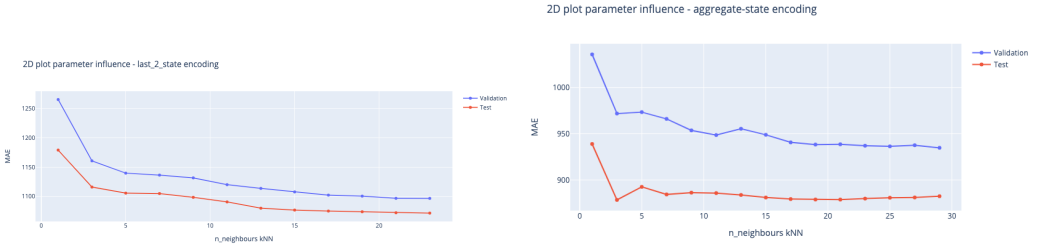
(a) 3D last-2-state encoding

(b) 3D aggregated encoding

5.2 K-nearest-neighbour regression

For the K nearest neighbour regression, the mean absolute error was measured when adjusting for the number of neighbours. This test also included how the test and validation subset and the separation of aggregated, last-2-state encoding were affected in the analysis. The analysis showed the steady decrease of the mean absolute error in both subsets when the number of neighbours increased. Both aggregated and last-2-state encoding datasets shown in Figure 6b and 6a showed this result to be true. These tests showed that in a range of 1 to 25, 23 was the optimal number of nearest neighbours when evaluated with the mean absolute error.

It is important to note that when evaluating the model, it is not only the reduction of variance that must be taken into account. The bias is also affected when the number of nearest neighbours is adjusted. As the number of nearest neighbours increases, the bias decreases because the model becomes more complex to adapt to the original dataset.



(a) kNN last-2-state encoding

(b) kNN aggregated encoding

5.3 Random forest regression

The last analysis implemented another regression algorithm, the random forest regression. This algorithm implemented feature optimization as a technique to help improve the mean absolute error score of the existing models. An overview of the results with the Random forest regression can be seen in table 4.

Parameters	Encoding	MAE	MSE	RMSE	R^2
Default	Last-2-state	1084.83	6961107.25	2638.39	0.39
Default	Agg-state	860.49	5516437.04	2348.71	0.52
Optimal	Last-2-state	1062.70	6967395.60	2639.58	0.39
Optimal	Agg-state	889.32	5772488.12	2402.60	0.50

Table 4: Overview of random forest regression model performance

To assess how the number features influenced the mean absolute error, a 2d plot was made. Figure 7 shows that with a range of 38 how the sparsely declined the mean absolute error as features increased. The range of features was calculated by the importance it served towards the algorithm.

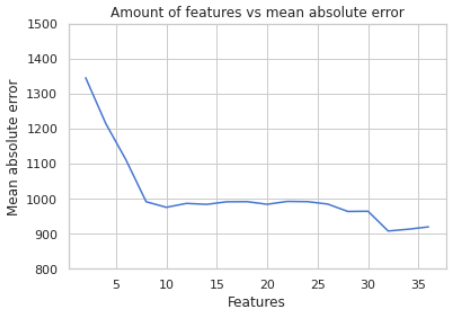


Figure 7: Features vs MAE

6 Conclusions

In summary, the goal of this assignment was for the regression tree, kNN, and random forest regression models to predict the the remaining time for each case. The data set provided was based on a real-life event log of sepsis cases from a hospital. The data set consisted of 39 attributes, with 5176 events recorded from 16 distinct activities, stemming from roughly 1000 cases. As the domain is unknown to us, possible missing values were dropped from the data set. The performance of each respective model and encoding combination was tested using various performance measures. Cross-validation was applied to evaluate how the models will perform on an independent data set. Parameters for each respective model were fine-tuned to find the settings that deliver best performance. Performance results showed that the regression tree, in combination with aggregated-state encoding, delivered the best performance for this data set. The regression tree model returned the lowest MAE score, which is the performance score we have chosen to decide which model was the best for this data set.

6.1 Implementation

To actually apply process optimization through machine learning, a regression tree model can be deployed online. The hospital's ERP-system can then use this. When such an implementation is desired, it is important to create a pipeline that treats all data in the same way. This pipeline can then ingest dynamic data to make not only historical predictions but also predictions in real time. With today's speed of progress in the medical world, it is also a consideration to apply continuous learning. The model will then be regularly retrained to respond to, for example, new treatments or a changing remaining time.

6.2 Future work

The current paper is an investigation into the prediction of the remaining time that a patient with sepsis will need to stay in the hospital. Possible future research can examine a broader clinical picture (other possible diseases, disorders or infections). In this way, such a system can be used for more departments. Future research can also examine the result of a medical procedure, such as inflammation values. When a certain medical procedure is performed on a patient, the expected improvement can be compared whether this corresponds to the actual improvement. In this way, the efficiency and effectiveness of the medical procedures can be examined and possibly improved.

7 Contributions of Group Members

7.1 Contributions of Group Members

Assignment 2 - Predictive Process Monitoring	**Team members**
Task 1: Exploring the data set	
1.1 Exploratory data analysis	Josh Bleijenberg
1.2 Data cleaning	Christian Acosta
1.3 Process Discovery and Visualization	Christian Acosta
Task 2: Preprocessing and Trace Encoding	
2.1 Trace Encoding	Sjoerd Vink
2.2 Create Training and Held-out test data sets	Sjoerd Vink
Task 3: Predicting Case Remaining Time - Regression Trees	Mischa van Ek
Task 4: Predicting Case Remaining Time - kNN Regression	Robin Wiersma
Task 5. Report your results and discuss your findings	Christian Acosta
Bonus Tasks	Josh Bleijenberg

Report	**Team members**
Abstract	Sjoerd Vink
1. Introduction	Josh Bleijenberg
2. Data	Robin Wiersma
3. Methods	Sjoerd Vink
4. Evaluation	Mischa van Ek
5. Discussion	Mischa van Ek
6. Overview and comparison of algorithms	Christian Acosta
7. Conclusion	Christian Acosta

7.2 Group reflection

Looking back on the progress of the project, this went well. During the first seminar, a clear division of tasks was drawn up with concrete deadlines. This allowed us to immediately start programming.

The second week, a meeting was held for the college to discuss progress. Unfortunately there was a miscommunication about the meeting so not everybody could attend the meeting. Nevertheless, all planned subtasks were performed as planned. Everyone adhered to the division of tasks, which made the progress of the project smooth.

Communication with the team went well. Direct communication took place via a WhatsApp group. A Kanban board has also been made on Microsoft Teams for the distribution of tasks. Here we could see not only our own tasks, but also those of others. During the project we took a close look at each other's results. There was also room to be critical, which improved the end result.

In short, the project went well and we are looking forward to assignment 3.

Appendices

A Process flow

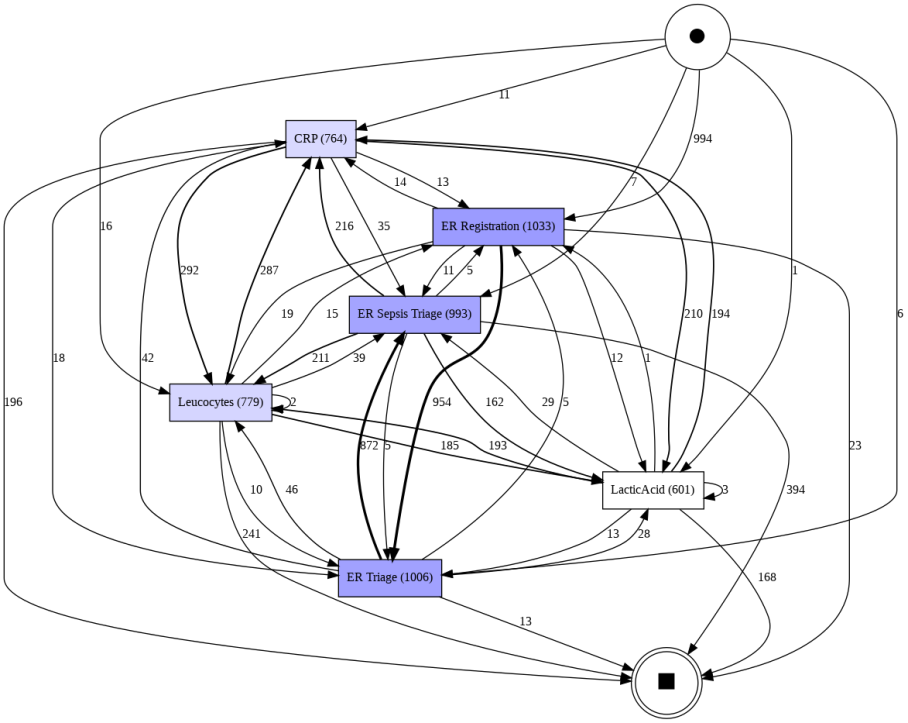


Figure 8: Directly-Follows Graph

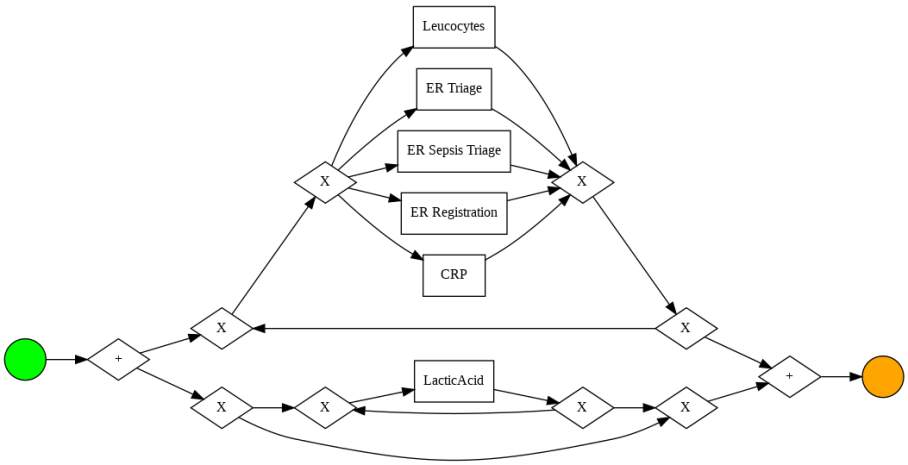


Figure 9: Business process model

B Decision tree

C Tree plot

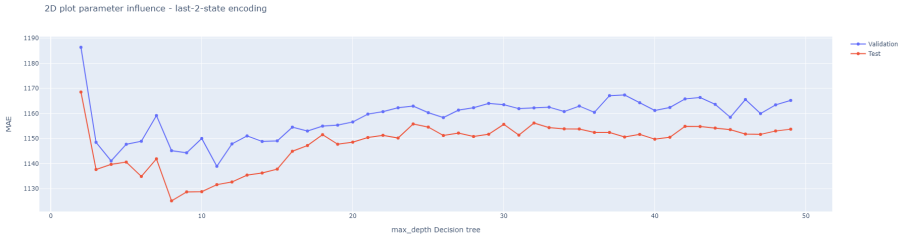


Figure 10: Longer range max depth

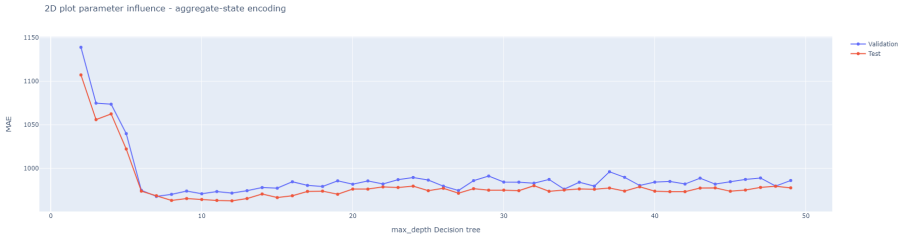


Figure 11: Longer range max depth

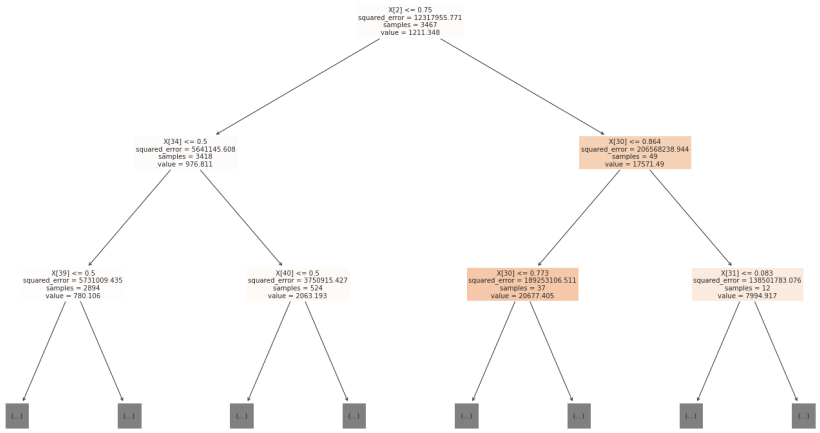


Figure 12: dt regressor - last-2-state

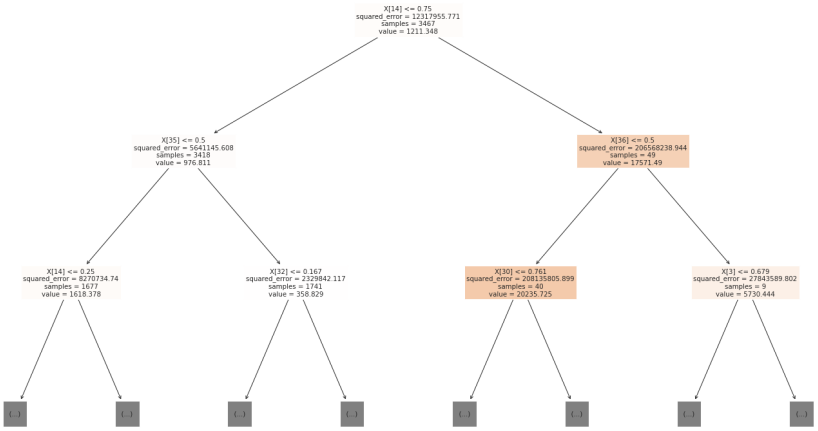


Figure 13: dt regressor - aggregated

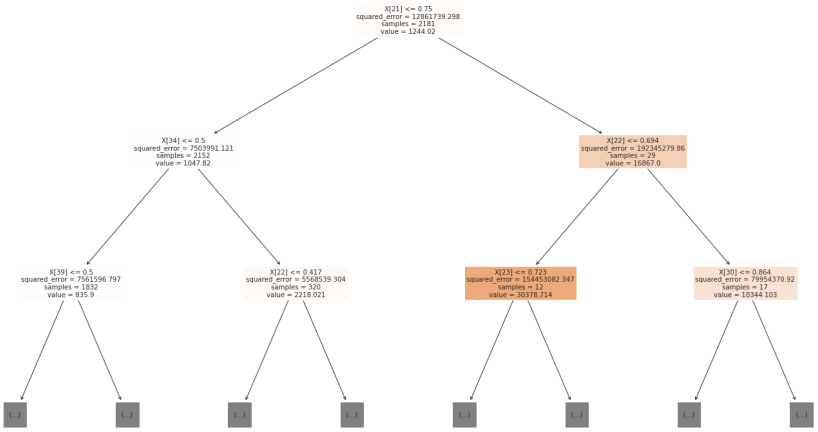


Figure 14: rf regressor - last-2-state

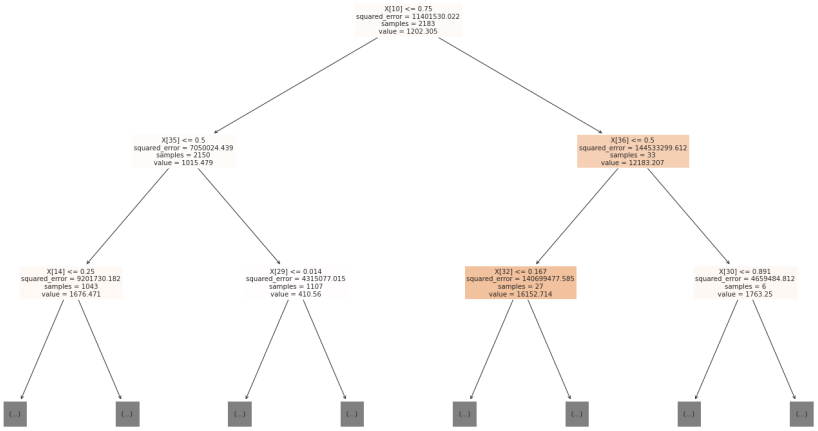


Figure 15: rf regressor - aggregated