# Introduction

In this Supplementary Information, we describe details on this study following chronological order of our analysis pipeline on systematic human learning using prelabor rupture of membranes (PROM) as an outcome. There are three of six sections corresponding to some sections in the main text, which are respectively Introduction, Software and equipment, and Procedure. Along with this PDF document, we also provide R Markdown (.Rmd) containing the same texts with this document but including the programming codes for the data analysis in-between of these texts. The R Markdown are available in <https://github.com/herdiantrisufriyana/shl>. To get raw data, one need to request an access from the BPJS Kesehatan for their sample dataset published in August 2019. Up to this date, there are three sample datasets they published in February 2019, August 2019, and December 2020. For the first and second versions, a request is applied via <https://e-ppid.bpjs-kesehatan.go.id/>, while the third is applied via <https://data.bpjs-kesehatan.go.id>. To preprocess the raw data into the input dataset of this study, follow the codes of the R Markdown in <https://github.com/herdiantrisufriyana/medhist/tree/main/preprocessing>.

# Software and equipment

We set up a programming environment for this study. Bioconductor was utilized as described in the main text. There were 198 R packages which are 9 base packages, 36 other packages, and 91 dependencies.

# Procedure

Here we applied systematic human learning, as described in the main text, to determine what were causal factors that can be inferred from our dataset.

## Step 1

We chose PubMed for a reason described in the main text. There is no source code for this step. One can choose >1 literature databases.

## Step 2

Keywords ‘“Fetal Membranes, Premature Rupture”[Mesh]’ were used to find a document from an authoritative institution. After reading that document, we wrote down the variable as a dataframe in this source code.

## Step 3 to 5

These steps are iteratives. During the these steps, we took note in this source code as a dataframe. We would process this dataframe to automatically construct a causal diagram in the next section.

## Step 6

In this step, we made a common-cause table for each causal factor. A unique set of the first-level factors were identified from the common-cause table. Then, we created tables of baseline edges and nodes before considering available data.

## Step 7

The data source was a sample dataset of the whole health insurance database during 2015 and 2016 by cross-sectional design. Stratified random sampling was applied. The strata variable was constructed from 66,072 combinations of all the healthcare facilities (*n*=22,024) and category of family, which were: (1) a family of which members never visit the healthcare facilities; (2) a family of which members have visited only primary care; and (3) a family of which members have visited all levels of care. For each stratum, one to ten families were randomly included. This means only 10 families were randomly included if more than that number, resulting 586,969 families with 1,697,452 subjects.

We conducted non-essential data cleaning, e.g. revising the inconsistent name of states, estimating the healthcare identifiers, *et cetera*. These procedures were parts of our R package of medhist 0.1.0. No sampling was conducted.

After the non-essential data cleaning, we applied retrospective cohort design, as described in the main text. For pregnant women, we use several codes for determining delivery or immediately after delivery care. The 220 codes are described.

We conducted data preprocessing after defining the target population and sampling it retrospectively. Demographics were included as categorical variables for causal inference. We also computed a number of days for a code in the latest encounter before the time of prediction, including those by codes as a causal factor.

We also determined the diagnosis and procedure codes that represented each causal factor. In this causal diagram, a measured causal factor was depicted, as described in the main text. Eventually, we recoded any code assigned to a causal factor into the name of that factor.

We need to ensure all inference using training set only. Data partition was conducted before continuing the downstream analysis. Therefore, causal inference did not use validation set.

## Step 8 to 9

We conducted causal inference as described in the main text. This will help us to include only the confirmed causal factors as candidate predictors before conducting pre-selection of those candidates to fulfill quality control of predictors in the main text. We included causal factors of which the data were available in training set. Details on this information and ICD-10 codes or demographical variables for each candidate of causal factors are shown in the next section.

## Step 10

All candidate predictors, including non-demographical causal factors, have non-zero variances. There were 460 candidate predictors fulfilling this criterion. We also showed in the same table that there are 426 candidate predictors without perfect separation.

We excluded the diagnosis/procedure codes that may leak the outcome information. We only used the existing codes in the training set to determine outcome-leaker codes based on the previous codes for determining delivery or immediately after delivery care. There were 54 codes that may leak the outcome. All of them were also irredundant.

By systematic human learning and causal inference using available data, we also determined causal factors as the candidate predictors. There were 27 first- and 10 second-level factors of PROM. Only data for 12 out of 27 causal factors were available in training set. Either the diagnosis/procedure codes, or demographical variables (not included as candidate predictors), for causal factors are also described.

Features were extracted on irredundant candidate predictors with non-zero variances and no perfect separation in training set only. The candidate predictors were transformed into binaries in all data partitions.

Previous data partition had not held out instances for calibration yet. This took 80% of training set. We also gave different weights for event and nonevent by including censored outcome For hyperparameter tuning, we applied 5-fold cross validation, instead of 10-fold as applied for PC modeling. Meanwhile, the final training and calibration for each model were conducted by bootstrapping for 30 times. The same resampling methods were applied for both classification and estimation tasks. We applied the tuning grids and the training configurations.