CGM data analysis report

Xinyu Dong

2026-01-18

Download data

The *CGMdata.csv* dataset is a perturbed and truncated toy dataset of Continuous Glucose Measurements based (rather loosely) off real-world readings of glucose levels of patients residing in the Emory healthcare system. In real world scenarios, though the patient is monitored closely during their occupancy in a hospital, due to technical reasons the glucose readings may be partially *missing*. Ideally we could predict the start of these missing intervals and perform data imputation as a next step.

Variables in the original dataset include:

- patient_id: the encoded patient id of a patient, patient id differs across patients and are categorical.
- glucose: a single numeric continuous glucose reading of a patient at time *t*, where t is indicated by the variable *time*.
- time: the time of the glucose reading in minutes from baseline observation.
- x1: an unknown feature variable in numerical form. Its value remains constant in patient, and could differ across patients.
- x2: an unknown feature variable in numerical form. Its value remains constant in patient, and could differ across patients.

For simplicity, this analysis focuses on variables patient_id, glucose, and time only. The research question of this analysis is: Can we use a reasonable statistical method to accurately predict the start of a missing interval in the glucose readings of patients? What is the accuracy of prediction?

Data Download:

```r
# first we load the required packages
library(here)
```

```
## here() starts at C:/Users/xdong63/OneDrive - Emory/Desktop/Fall 2024 Emory/BIOS 731/BIOS 731 homework
```

```r
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(scales)
```

```
## Warning: package 'scales' was built under R version 4.4.3
```

```r
library(tidyr)
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.4.3
```

```r
# then we read in the data using a "here" (relative path statement)
CGM_data <- read.csv(here("data", "CGMdata.csv"))
CGM_data <- CGM_data[, 1:3]
```

Data Cleaning

Now that we downloaded the continuous glucose monitoring data and filtered out the variables of interest, we need to fill in the missing time stamps first: glucose level for all missing timestamps are set to be zero as an indication of unobserved data, and we create a new indicator variable *ind* to indicate whether the data is observed (value 1) or missing (value 0).

Another indicator variable *start_ind* serves as the parameter of interest, which indicate whether the data is the start of a missing interval (value 1) or not (value 0). The time gap between two lines of data within the same individual is enforces to be 5 (minutes), which serves as the criteria for whether we should "insert" a line of data between two observed readings or not.

```r
if(!file.exists((here::here("data", "CGMdata_clean.csv")))){
  source(here("data", "data_cleaning.R"))
}else{
  readr::read_csv(here::here("data", "CGMdata_clean.csv"),
  show_col_types = FALSE)
}
```

```
## # A tibble: 26,365 x 5
##    patient_id  glucose  time   ind start_ind
##    <chr>         <dbl> <dbl> <dbl>     <dbl>
##  1 EM008_extra     253     0     1         0
##  2 EM008_extra     253     5     1         0
##  3 EM008_extra     257    10     1         0
##  4 EM008_extra     263    15     1         0
##  5 EM008_extra     244    20     1         0
##  6 EM008_extra     225    25     1         0
##  7 EM008_extra     224    30     1         0
##  8 EM008_extra     231    35     1         0
##  9 EM008_extra     235    40     1         0
## 10 EM008_extra     243    45     1         0
## # i 26,355 more rows
```

Data Modeling

Let Y_i denote the binary outcome for observation i, where

$$Y_i = \begin{cases} 1, & \text{if a start event occurs } (\texttt{start\_ind} = 1) \\ 0, & \text{otherwise } (\texttt{start\_ind} = 0). \end{cases}$$

Let $\mathbf{x}_i = (1, x_{i1}, x_{i2}, \ldots, x_{ip})^\top$ be the vector of predictors for observation i, including an intercept term. In this analysis, all available feature variables are considered as predictors except `ind, x1, x2`, which are excluded from modeling.

Model specification

We model the conditional probability of a start event using a binary logistic regression:

$$\pi_i \equiv \Pr(Y_i = 1 \mid \mathbf{x}_i),$$

with the logit link function

$$\log\left(\frac{\pi_i}{1-\pi_i}\right) = \mathbf{x}_i^\top \boldsymbol{\beta} = \beta_0 + \sum_{j=1}^{p} \beta_j x_{ij}.$$

Equivalently, the predicted probability is

$$\pi_i = \frac{\exp(\mathbf{x}_i^\top \boldsymbol{\beta})}{1 + \exp(\mathbf{x}_i^\top \boldsymbol{\beta})}.$$

Estimation

Parameters $\boldsymbol{\beta} = (\beta_0, \beta_1, \ldots, \beta_p)^\top$ are estimated by maximum likelihood. Under independence across observations, the likelihood is

$$L(\boldsymbol{\beta}) = \prod_{i=1}^{n} \pi_i^{Y_i} \left(1 - \pi_i\right)^{1-Y_i},$$

and the log-likelihood is

$$\ell(\boldsymbol{\beta}) = \sum_{i=1}^{n} \left[Y_i \log(\pi_i) + (1 - Y_i)\log(1 - \pi_i)\right].$$

The maximum likelihood estimate $\widehat{\boldsymbol{\beta}}$ is obtained by numerically maximizing $\ell(\boldsymbol{\beta})$.

Train/validation evaluation

To assess generalization performance, we split the dataset into a training set and a validation set. Model parameters are estimated using only the training data, and predictions and performance metrics are computed on the held-out validation data. This procedure reduces optimistic bias compared to evaluating on the same data used for model fitting.

Prediction and classification

For each observation i, we compute the fitted probability

$$\widehat{\pi}_i = \frac{\exp(\mathbf{x}_i^\top \widehat{\boldsymbol{\beta}})}{1 + \exp(\mathbf{x}_i^\top \widehat{\boldsymbol{\beta}})}.$$

To obtain a hard class label, we apply a decision threshold $\tau = 0.5 \in (0,1)$:

$$\widehat{Y}_i = \begin{cases} 1, & \text{if } \widehat{\pi}_i \geq \tau, \\ 0, & \text{if } \widehat{\pi}_i < \tau. \end{cases}$$

Using the predicted labels $\widehat{Y}_i$ and true labels Y_i, we summarize performance with a confusion matrix and compute accuracy:

$$\text{Accuracy} = \frac{1}{n}\sum_{i=1}^{n} \mathbb{I}\left(\widehat{Y}_i = Y_i\right),$$

where $\mathbb{I}(\cdot)$ is the indicator function.

```r
if(!file.exists((here::here("data", "CGMdata_pred.csv")))){
  source(here("modeling", "CGM_data_modeling.R"))
}else{
  readr::read_csv(here::here("data", "CGMdata_pred.csv"),
  show_col_types = FALSE)
}
```

```
## # A tibble: 7,910 x 7
##    patient_id  glucose  time   ind start_ind pred_prob pred_class
##    <chr>         <dbl> <dbl> <dbl>     <dbl>     <dbl>      <dbl>
##  1 EM008_extra     253     0     1         0  2.22e-16          0
##  2 EM008_extra     257    10     1         0  2.22e-16          0
##  3 EM008_extra     263    15     1         0  2.22e-16          0
##  4 EM008_extra     244    20     1         0  2.22e-16          0
##  5 EM008_extra     235    40     1         0  2.22e-16          0
##  6 EM008_extra     243    45     1         0  2.22e-16          0
##  7 EM008_extra     233    65     1         0  2.22e-16          0
##  8 EM008_extra     220   100     1         0  2.22e-16          0
##  9 EM008_extra     220   105     1         0  2.22e-16          0
## 10 EM008_extra     215   140     1         0  2.22e-16          0
## # i 7,900 more rows
```

```r
CGM_data_pred <- readr::read_csv(here::here("data", "CGMdata_pred.csv"),
  show_col_types = FALSE)

if (!exists("accuracy_val") || !exists("cm")) {
  accuracy_val <- mean(CGM_data_pred$pred_class == CGM_data_pred$start_ind, na.rm = TRUE)
  cm <- table(Truth = CGM_data_pred$start_ind, Prediction = CGM_data_pred$pred_class)
}

cat("\nValidation accuracy:", round(accuracy_val, 4), "\n\n")
```

```
##
## Validation accuracy: 0.9987
```

```r
print(cm)
```

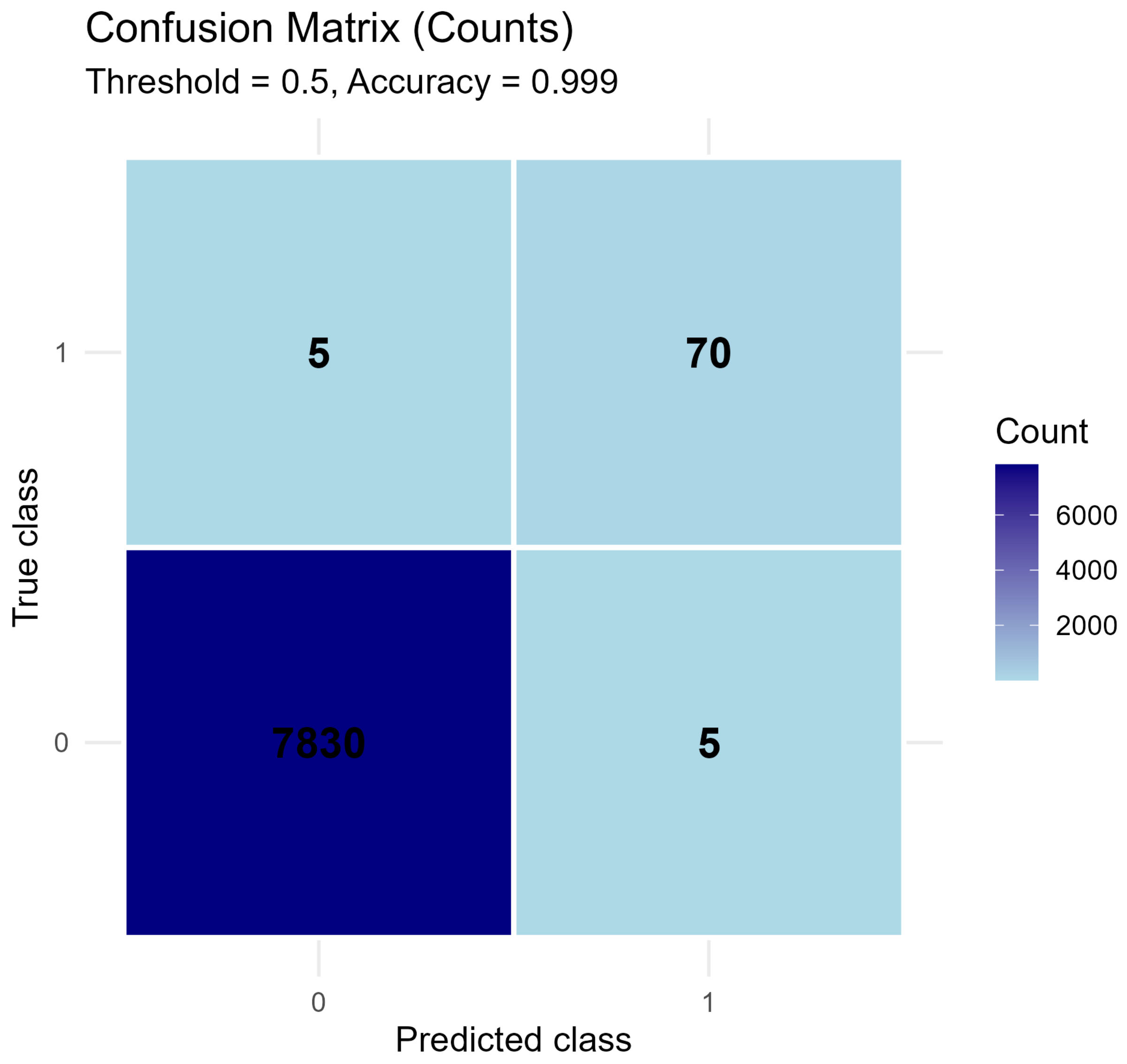
```
##      Prediction
## Truth    0    1
##     0 7830    5
##     1    5   70
```

Visualization

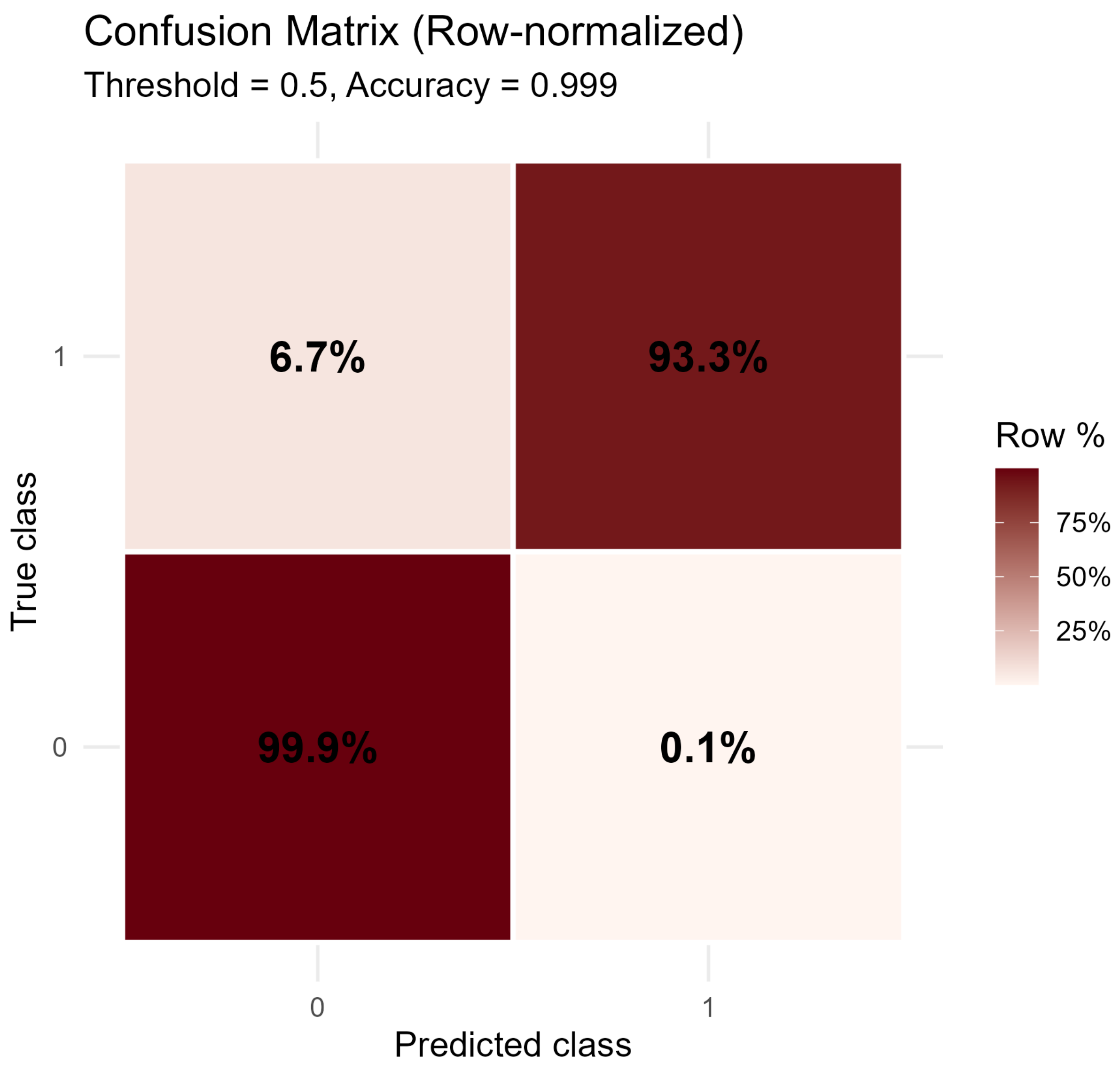
We perform a visualization of the confusion matrix obtained in section **Modeling**. The first plot shows a confusion matrix by raw counts, and the second shows a row normalized confusion matrix.

```
source(here("visualization", "CGM_data_visualization.R"))
```

```
knitr::include_graphics(
  here("results", "confusion_matrix_counts.png")
)
```



```
knitr::include_graphics(
  here("results", "confusion_matrix_row_normalized.png")
)
```


Confusion Matrix (Row-normalized)
Threshold = 0.5, Accuracy = 0.999
6.7%
93.3%
99.9%
0.1%
1
0
True class
0
1
Predicted class
Row %
75%
50%
25%