This is title

Feng Gu(T00751197), author 2 and author 3

April 15, 2025

Abstract Here to attach abstract.

1 Spline improved Logistic regression

In this section, we will try to create a Bayesian improved logistic regression using splines. The ideas is to use the splines to model the non-linear relationship between the variables and the response variables. Instead of using the classical statistical methods - Maximum Likelihood Estimation (MLE) to estimate the parameters of the model, we will use the Bayesian approach to estimate the posterior distribution of the parameters.

This section is primarily split into two parts:

- Part one: The theoretical background and practical implementation of the Bayesian improved logistic regression using splines.
- Part two: the module built from scratch to implement and customize the Bayesian improved logistic regression using splines.

Waiting to be added.

1.1 The likelihood function of β in the logistic regression

To a data set having target variable that has only two vlaues, we view the target variable follows a Bernoulli distribution with true parameter p. The parameter is the probability of the target variable being 1 and is determined by the linear combination of the input variables X and the parameters β. It gives: $Y_i \sim Bernoulli(p_i)$.

By linking function - $log(\frac{x}{1-x})$, the conditional probability of the target variable Y given the input variables X is given by [1]:

$$log(\frac{p}{1-p}) = X\beta,$$

Where p is the vector of true predicted probabiltiy of Y being 1, X is design matrix of the input variables, and β is the vector of the parameters.

$$p = \begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_n \end{bmatrix}, \quad X = \begin{bmatrix} X_{11} & X_{12} & \cdots & X_{1p} \\ X_{21} & X_{22} & \cdots & X_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ X_{n1} & X_{n2} & \cdots & X_{np} \end{bmatrix}, \quad \beta = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{bmatrix}.$$

After taking the inverse of the link function, we can get the predicted probability of the target variable being 1 [2]:

$$p = \sigma(X, \beta) = \frac{1}{1 + e^{-X\beta}}.$$

Based on the above probability model, we can get the likelihood function of parameters β given the data set (X, Y):

$$L(\beta|X, Y) = \prod_{i=1}^{n} p_i^{Y_i}(1 - p_i)^{1-Y_i} \tag{1-1}$$

Taking logarithm of the likelihood function and replacing p_i with $\sigma(X_i, \beta)$[3], we can get the log-likelihood function of the parameters β given the data set (X, Y):

$$\begin{aligned} \ell_n(\beta) &= \sum_{i=1}^{n} [y_i log(\sigma(x_i, \beta)) + (1 - y_i) log(1 - \sigma(x_i, \beta))] \\ &= \sum_{i=1}^{n} (x_i y_i \beta - log(1 + e^{x_i \beta})) \end{aligned}$$

[1] $\frac{p}{1-p}$ is doing broadcast operation, which makes sure the result is still a vector not linear algebra dot product.

[2] 'σ' is a sigmoid function, which takes X and β as input and returns the predicted probability, we will use it later for simplicity.

[3] X_i is the input vector of ith observation.

We search for the maximum of the log-likelihood function to get the MLE of the parameters β:

$$\hat{\beta} = \arg\max_{\beta} \ell_n(\beta)$$

To find a good solution for the MLE, we can use the Newton-Raphson method to iteratively update the parameters β:

$$\beta_{k+1} = \beta_k - H^{-1}(\beta_k)\nabla\ell_n(\beta_k)$$

Where $H(\beta_k)$ is the Hessian matrix of the log-likelihood function $\ell_n(\beta)$ at β_k, and $\nabla\ell_n(\beta_k)$ is the gradient vector of the log-likelihood function $\ell_n(\beta)$ at β_k. The initial value of β can be set to any proper value, but a good choice can accelerate the convergence speed.

The Hessian matrix and the gradient vector can be calculated as follows [4]:

$$H(\beta) = -\sum_{i=1}^{n} \sigma(x_i, \beta)(1 - \sigma(x_i, \beta))x_i x_i^T$$

$$\nabla\ell_n(\beta) = \sum_{i=1}^{n} (y_i - \sigma(x_i, \beta))x_i$$

1.2 The estimate of prior distribution of β

1.3 Transfomation on design matrix for regression splines

Regression splines is a powerful tool to model the non-linear relationship between the input variables and the response variable. It splits the input space into several intervals and fits basis functions to each interval. The basis functions can be choosen to be simple linear or polynomial functions, or more complex functions. The functions across the intervals are connected at the knots, which makes sure the total regression function is continuous and smooth.

Given a univariate predictor $x \in R^n$, we can construct a regression spline design matrix by transforming x into a set of basis functions. For a spline of degree d with K knots $\{\xi_1, \xi_2, \ldots, \xi_K\}$, the new design matrix $\mathbf{X}_{\text{spline}}$ is:

$$\mathbb{K} : X(n, p) \mapsto X_{spline}(n, p + C), \quad C > 1$$

[4] We assume the $X's$ are independent that we can add n individual observed Fisher information to get the total Fisher information. H is summation of n matrices in (p, p) , $\nabla\ell_n(\beta)$ is the vector of the first derivative of the log-likelihood function

$$\mathbf{X}_{\text{spline}} = \begin{bmatrix} 1 & x_{11} & x_{11}^2 & x_{21} & x_{21}^2 & \cdots & x_{p1}^d & (x_{11}-\xi_{11})_+^d & \cdots & (x_{p1}-\xi_{pK})_+^d \\ 1 & x_{12} & x_{12}^2 & x_{22} & x_{22}^2 & \cdots & x_{p2}^d & (x_{12}-\xi_{11})_+^d & \cdots & (x_{p2}-\xi_{pK})_+^d \\ \vdots & \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots & & \vdots \\ 1 & x_{1n} & x_{1n}^2 & x_{2n} & x_{2n}^2 & \cdots & x_{pn}^d & (x_{1n}-\xi_{11})_+^d & \cdots & (x_{pn}-\xi_{pK})_+^d \end{bmatrix}$$

Here, $(x-\xi_j)_+^d$ denotes the truncated power basis function:

$$(x-\xi_j)_+^d = \begin{cases} (x-\xi_j)^d & \text{if } x > \xi_j \\ 0 & \text{otherwise} \end{cases}$$

This design matrix allows the regression model to fit a flexible, piecewise polynomial function with continuity at the specified knots.

1.4 Module design and implementation in Python

We use Python to achieve the above calculation by using functions provided by the libraries that include: Numpy, Scipy (waiting to be added).