The Logistic Regression model - Estimation and Inference

M2 D3S/EGR 2025-2026

Louis Olive

louis.olive@gmail.com / louis.olive@ut-capitole.fr

September 24, 2025

Outline

Outline

- The Logistic Regression model
- Estimation
- Inference
- Goodness of Fit

The Logistic Regression model

The Logistic Regression model

Informal introductory example

We provide here some intuitions leading to the Logistic Regression model using a simulated data set from James et al. (2021) (the **Default** data set).

Code

```r
# Default data set (simulated) from ESLII/ISLR
default_data <- ISLR2::Default %>%
    as_tibble()

glimpse(default_data)
```

```
Rows: 10,000
Columns: 4
$ default <fct> No, No, No, No, No, No, No, No, No, No, No, No, No, No, No, No…
$ student <fct> No, Yes, No, No, No, Yes, No, Yes, No, No, Yes, Yes, No, No, N…
$ balance <dbl> 729.5265, 817.1804, 1073.5492, 529.2506, 785.6559, 919.5885, 8…
$ income  <dbl> 44361.625, 12106.135, 31767.139, 35704.494, 38463.496, 7491.55…
```

This is a toy data set used for teaching purposes containing information on ten thousand customers.

The aim here is to assess which customers will ***default*** on their credit card debt (the target or response variable) based on the current credit card ***balance*** and other individual characteristics (the predictors or feature vector).

The Logistic Regression model

Informal introductory example

We can start to explore the Default data with a scatterplot (Figure 1) of the target variable (***default***) with respect to a predictor (***balance***):

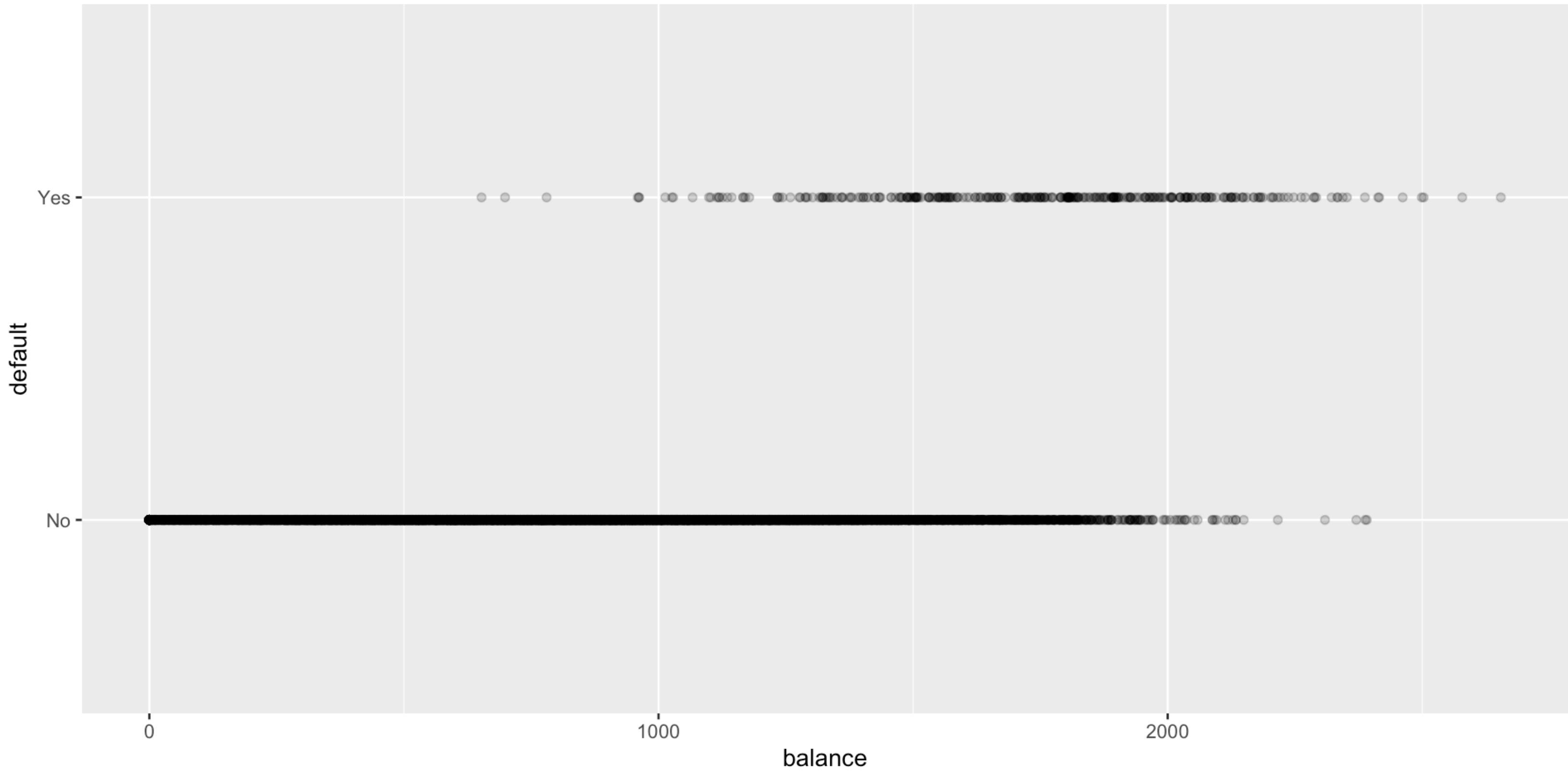
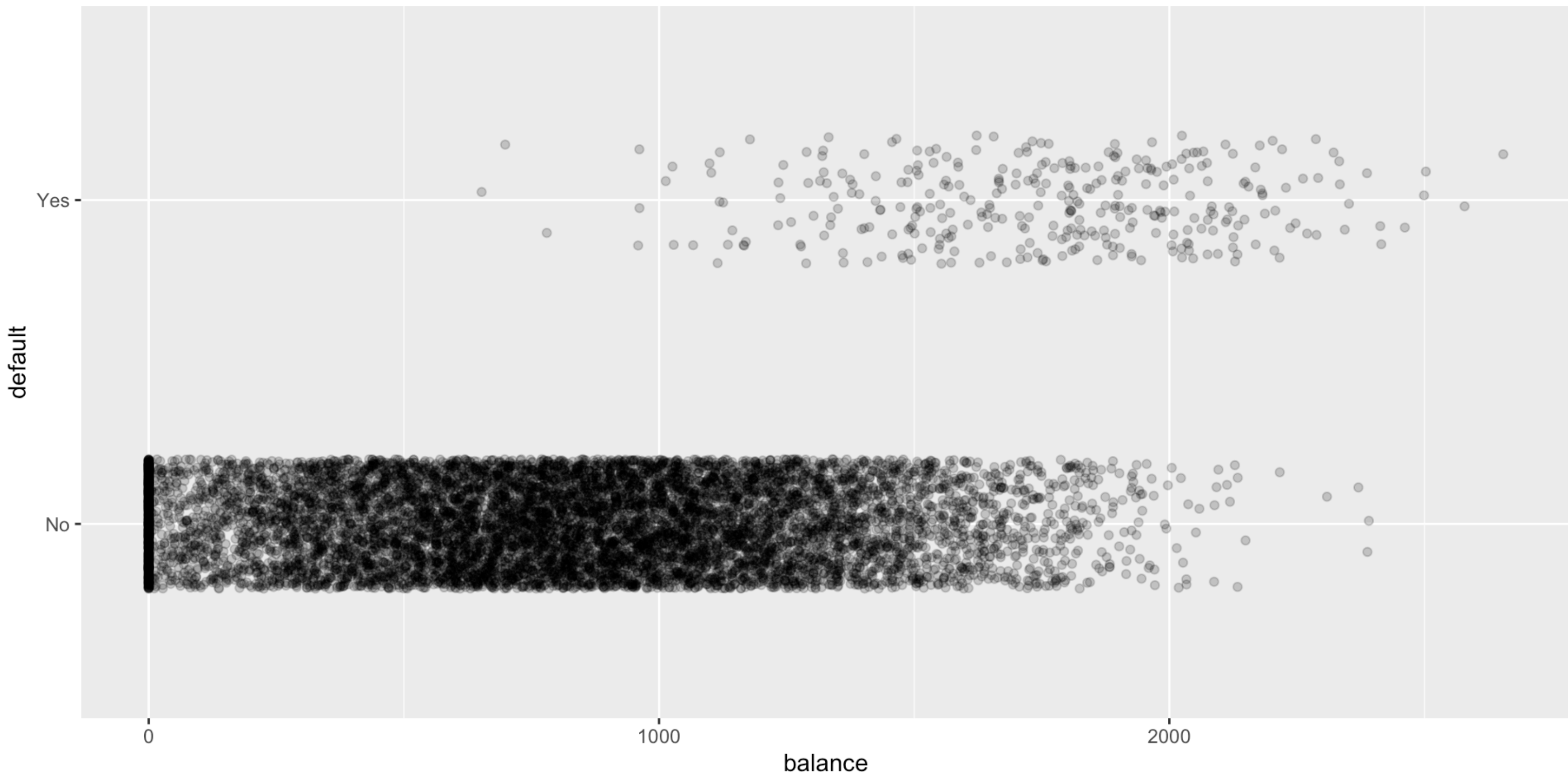


Figure 1: Scatterplot of variable ***default*** with respect to credit card ***balance*** for 10000 customers

The Logistic Regression model

Informal introductory example

In this scatterplot, all points fall on one of two parallel lines representing the absence (No) or occurrence (Yes) of ***default***. We "jitter" the data vertically to avoid overplotting. The plot below shows that the response variable is imbalanced towards the absence of default:



The Logistic Regression model

Informal introductory example

We also show the boxplots of credit cards ***balance*** with respect to ***default*** status:

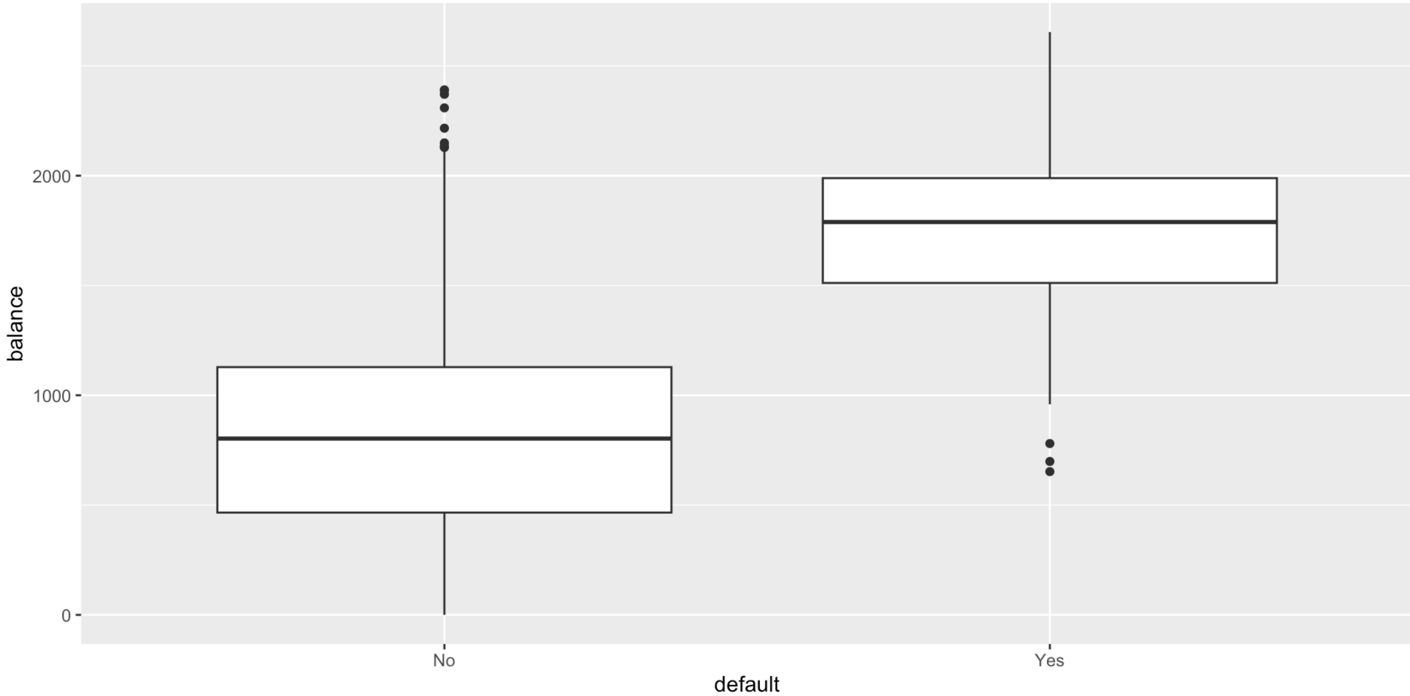


Figure 2: Variable ***balance*** with respect to ***default*** status

We can see from Figure 1 and Figure 2 that default tends to be more prevalent for accounts with a high balance. However it is difficult to guess a simple relationship between default and balance.

The Logistic Regression model

Informal introductory example

To investigate further we discretise the balance variables by classes of width $300\$$ and compute the mean of response variable (***default*** is Yes) within each balance class:

```
# A tibble: 9 × 6
  balance_bins   min   max    No   Yes `Mean(default)`
  <fct>        <dbl> <dbl> <int> <int>           <dbl>
1 [0,300)          0   300  1497     0          0
2 [300,600)      300   600  1784     0          0
3 [600,900)      600   900  2305     3          0.0013
4 [900,1200)     900  1200  2098    19          0.009
5 [1200,1500)   1200  1500  1330    53          0.0383
6 [1500,1800)   1500  1800   527    96          0.154
7 [1800,2100)   1800  2100   115   114          0.498
8 [2100,2400)   2100  2400    11    41          0.788
9 [2400,2700)   2400  2700     0     7          1
```

The Logistic Regression model

Informal introductory example

Then we plot the mean of default (in red) within each balance class (of width $300\$$):

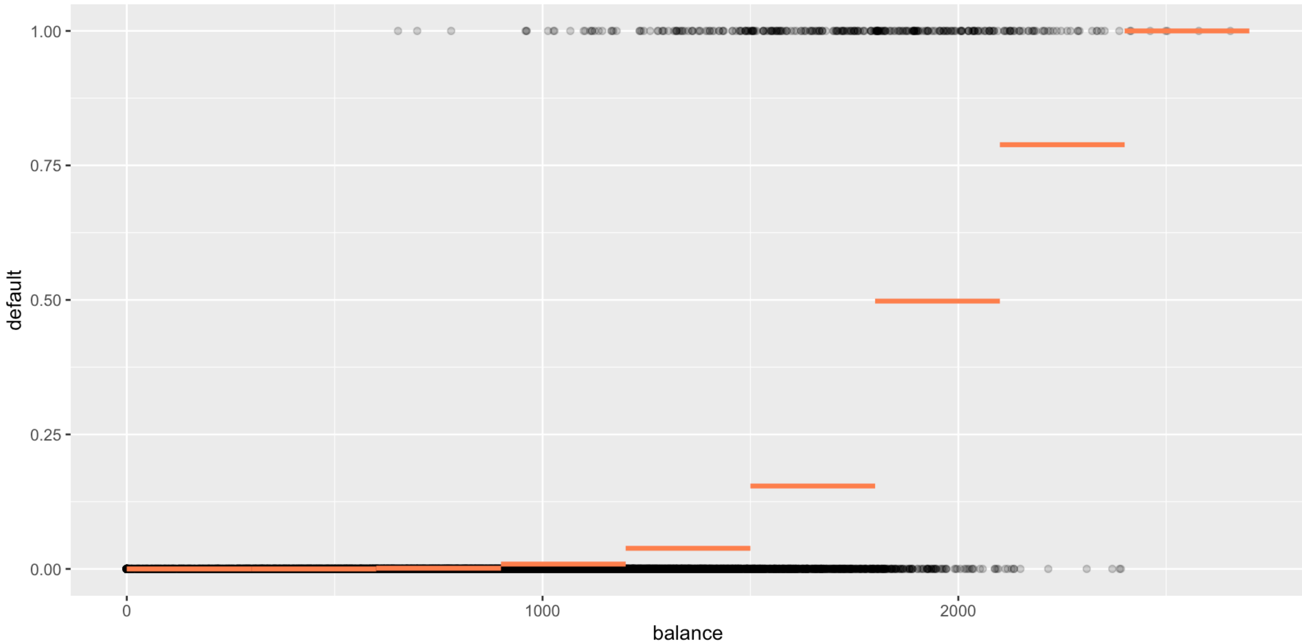


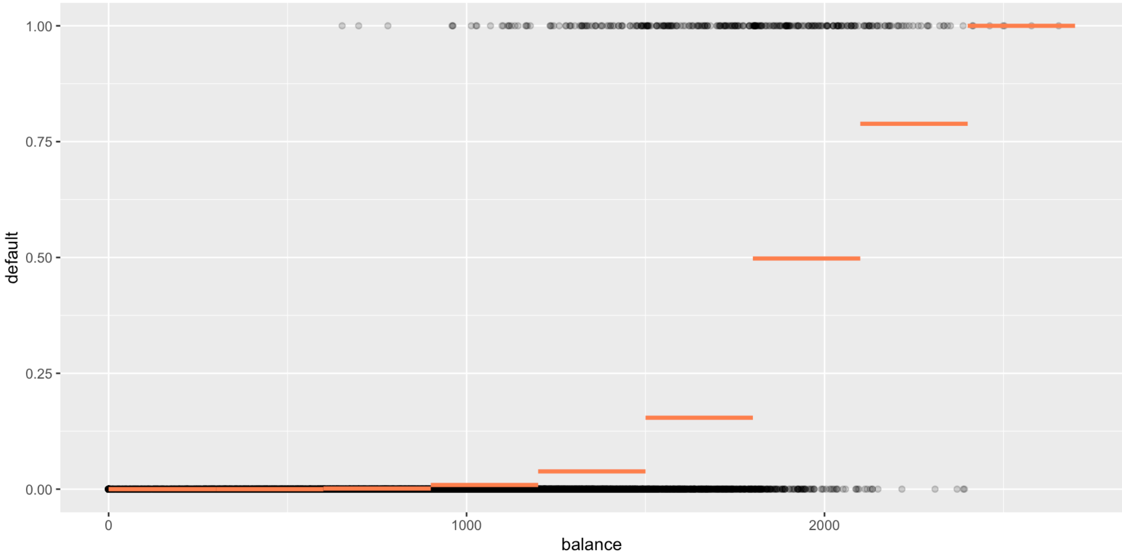
Figure 3: Mean occurrence of ***default*** within ***balance*** classes

The relationship between the mean occurrence of ***default*** and ***balance*** is easier to read.

Figure 3 clearly shows that as balance increases, the proportion of customers defaulting on their credit card increases.

The Logistic Regression model

Informal introductory example



We also notice that the mean default occurrence with respect to balance classes follows a kind of "S"-shaped curve or **sigmoid** function. Going further and informally, considering that the mean of default occurrence is an estimate of $\mathbf{E}[Y|X = x]$ for each balance classes an idea would be to model:

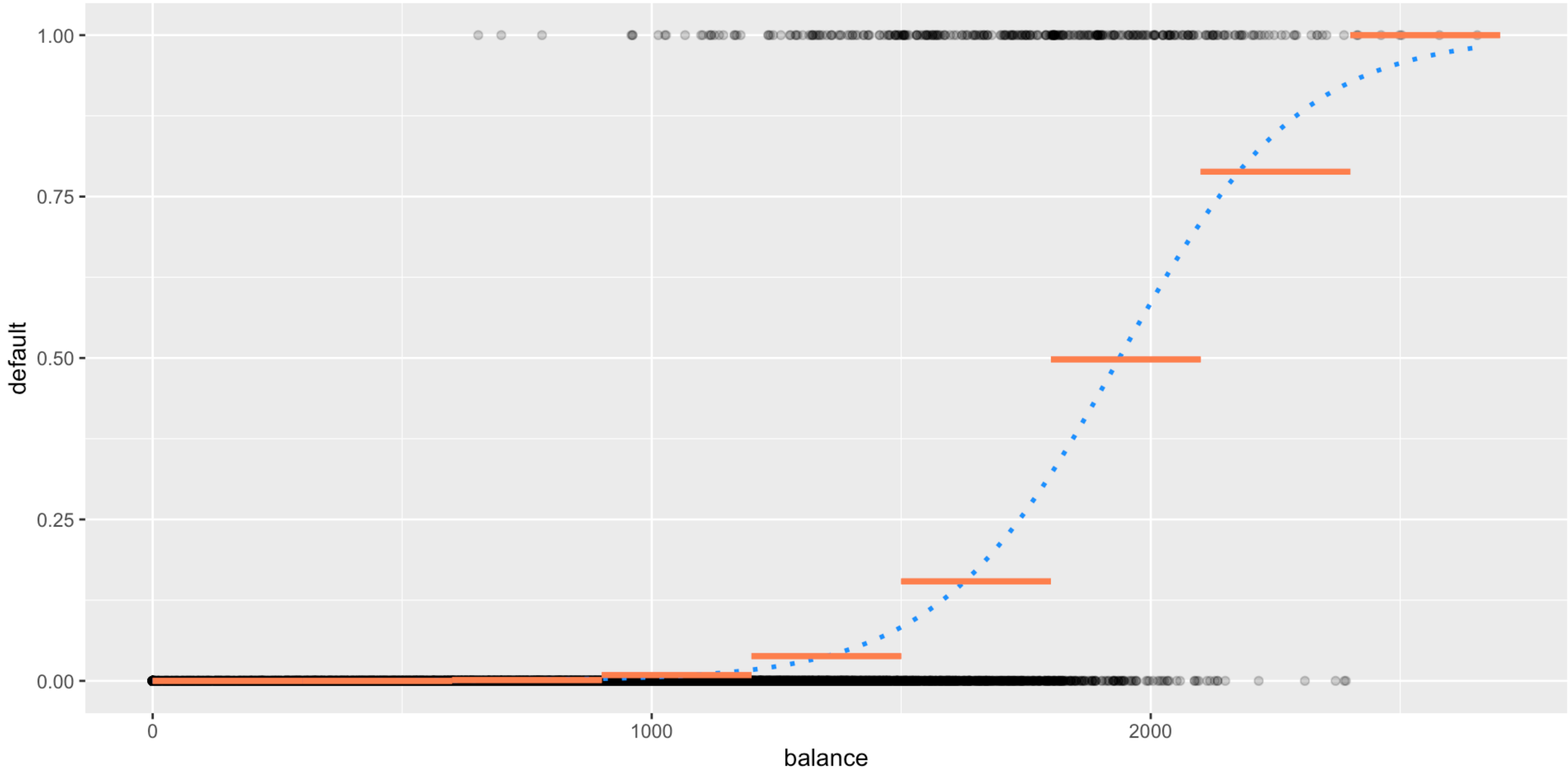
$$\mathbb{E}[Y|X = x] = \mu_\beta(x)$$

where μ_β is a **sigmoid** function in $[0, 1]$.

The Logistic Regression model

Informal introductory example

The Logistic Regression model uses the **sigmoid** function $\sigma : x \to \sigma(x) = \frac{e^x}{1+e^x}$ also known as the logistic function. Below a simple transform of this logistic function has been "fitted" (blue dots) to the `default ~ balance` data set:



The Logistic Regression model

A more formal definition - the discriminative approach

Reminding the statistical learning / scoring concepts introduced before. We try to predict the output `default` ($Y \in \{0, 1\}$) using a training set of inputs X: this is a binary classification problem.

We remind that to estimate an optimal classifier for output $Y \in \{0, 1\}$ using input $X = (X_1, \cdots, X_p)$ one approach was to:

- model the conditional distribution $Y|X$ (the **discriminative** approach),
- estimate $\eta(x)$ with:

$$\eta(x) = \mathbb{P}[Y = 1|X = x)] = \mathbb{E}[Y = 1|X = x)]$$

- $\eta(x)$ can be used as a Scoring function
- define the classifier f_s using a cutoff s and Scoring function $\eta(x)$ to predict output Y:

$$f_s(x) = \begin{cases} 1 & \text{if } \eta(x) \geq s \\ 0 & \text{otherwise} \end{cases}$$

The Logistic Regression model

A more formal definition

In the case of **Logistic Regression** classifier, we model:

$$Y|X = x \ \sim B(\eta(x))$$

with

$$\eta(x) = \sigma(x^T\beta) = \frac{exp(x^T\beta)}{1 + exp(x^T\beta)}$$

for some parameter $\beta = (\beta_1, \cdots, \beta_p) \in \mathbb{R}^p$, usually $x_1 = 1$ and β_1 is an intercept. σ is the sigmoid logistic function we have seen before.

In the literature is usual to denote $\eta(X) = p_\beta(X)$ or $\eta(X) = \pi_\beta(X)$.

From now, we will use the notation $p_\beta(X)$.

The Logistic Regression model

A more formal definition

Defining $\mathbf{logit} : x \rightarrow \log\left(\frac{x}{1-x}\right)$, which is the inverse of $\boldsymbol{\sigma}$ the logistic function (show it as an exercise), we have:

$$\mathrm{logit}(p_\beta(X)) = X^T\beta$$

The Logistic Regression model:

We are given $(x_i, y_i) \in \mathbb{R}^p \times \{0, 1\}$, $i = 1, \cdots, n$

The Logistic Regression model assumes that outputs $\boldsymbol{y_i}$ are independent Bernoulli with parameter $\boldsymbol{p_\beta(x_i)}$ depending on $\boldsymbol{x_i}$:

$$\mathrm{logit}(p_\beta(x_i)) = x_i^T\beta$$

Estimation

Estimation

How to fit with R

The syntax to fit the Logistic model in R using `glm()` is:

```
glm(y ~ x, data = dataframe, family = binomial(link = ''logit''))
```

The formula $\mathbf{y} \sim \mathbf{x}$ depicts the model (i.e. inputs are X, output is Y) and the `data=` argument points to the training set contained in a R dataframe (or tibble). This is quite similar to the `lm()` function.

We also need to specify the distribution for the conditional Y values (binomial) and the link function (logit) via the `family=` argument.

Estimation

How to fit with R

For our default example:

The command `summary` produces result summaries of the fitted model:

```
Call:
glm(formula = default ~ ., family = "binomial", data = default_data)

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.087e+01  4.923e-01 -22.080  < 2e-16 ***
studentYes  -6.468e-01  2.363e-01  -2.738  0.00619 **
balance      5.737e-03  2.319e-04  24.738  < 2e-16 ***
income       3.033e-06  8.203e-06   0.370  0.71152
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2920.6  on 9999  degrees of freedom
Residual deviance: 1571.5  on 9996  degrees of freedom
AIC: 1579.5

Number of Fisher Scoring iterations: 8
```

We will see in the next lesson, how this model is fitted in practice and how to interpret or understand what is printed by the `summary` function.

Estimation

Maximum Likelihood Estimator

We are given $(x_i, y_i) \in \mathbb{R}^p \times \{0,1\}$, $i = 1, \cdots, n$ where outputs y_i are independent Bernoulli with parameter $p_\beta(x_i)$ depending on x_i:

$$\text{logit}(p_\beta(x_i)) = x_i^T \beta$$

The parameters β of the Logistic Regression model are usually determined using Maximum Likelihood Estimation (MLE). It consists on finding β for which the joint probability of the observed data is greatest.

As y_i are independent the likelihood function (joint probability) is the product of the probability mass functions:

$$L(Y, \beta) = \prod_{i=1}^{n} p_\beta(x_i)^{y_i} (1 - p_\beta(x_i))^{1-y_i}$$

with $Y = (y_1, \cdots, y_n)$ and $\beta = (\beta_1, \cdots, \beta_p)$.

Estimation

Maximum Likelihood Estimator

We seek to maximize the likelihood function over β, it is equivalent but easier to maximize the log-likelihood:

$$\begin{aligned}
\ell(Y,\beta) = \log L(Y,\beta) &= \sum_{i=1}^{n} \left(y_i \log(p_\beta(x_i)) + (1-y_i)\log(1-p_\beta(x_i))\right) \\
&= \sum_{i=1}^{n} \left(y_i \log\left(\frac{p_\beta(x_i)}{1-p_\beta(x_i)}\right) + \log(1-p_\beta(x_i)) \right) \\
&= \sum_{i=1}^{n} \left(y_i x_i^T \beta - \log(1+\exp(x_i^T\beta)) \right)
\end{aligned}$$

If the MLE $\hat{\beta}$ exists, the gradient of log-likelihood satisfies (first order necessary condition):

$$\nabla \ell(Y,\beta) = \left(\frac{\partial \ell(Y,\beta)}{\partial \beta_1}, \cdots, \frac{\partial \ell(Y,\beta)}{\partial \beta_p} \right) = \mathbf{0}$$

Estimation

Maximum Likelihood Estimator

We have for $j = 1, \cdots, p$:

$$\frac{\partial \ell(Y,\beta)}{\partial \beta_j} = \sum_{i=1}^{n} \left(y_i x_{ij} - x_{ij} \frac{\exp(x_i^T \beta)}{1 + \exp(x_i^T \beta)} \right) = \sum_{i=1}^{n} x_{ij} \left(y_i - p_\beta(x_i) \right)$$

In vector form:

$$\nabla \ell(Y,\beta) = \sum_{i=1}^{n} x_i \left(y_i - p_\beta(x_i) \right) = X^T (Y - P_\beta)$$

where:

$$X = \begin{pmatrix} x_{11} & \cdots & x_{1p} \\ x_{21} & \cdots & x_{2p} \\ \vdots & \vdots & \vdots \\ x_{n1} & \cdots & x_{np} \end{pmatrix} = \begin{pmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_n^T \end{pmatrix} \in \mathbb{R}^{n \times (p)}, \quad Y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \quad and \quad P_\beta = \begin{pmatrix} p_\beta(x_1) \\ p_\beta(x_2) \\ \vdots \\ p_\beta(x_n) \end{pmatrix}$$

Estimation

Maximum Likelihood Estimator

In the literature $\nabla \ell(Y, \beta)$ is denoted as the Fisher's score function $S(\beta)$.

If the MLE $\hat{\beta}$ exists, we have:

$$S(\hat{\beta}) = \nabla \ell(Y, \hat{\beta}) = X^T(Y - P_{\hat{\beta}}) = 0$$

Solving this equation involves solving p non-linear equations in $\beta = (\beta_1, \cdots, \beta_p)$:

$$y_1 x_{1j} + \cdots + y_n x_{nj} = x_{1j} \frac{\exp(x_1^T \beta)}{1 + \exp(x_1^T \beta)} + \cdots + x_{nj} \frac{\exp(x_n^T \beta)}{1 + \exp(x_n^T \beta)}, \quad j = 1, \cdots, p$$

Numerical methods are used to solve these non-linear equations as no closed-form solution exist.

Estimation

MLE existence

If we assume that $rank(X) = p$, we have that $S(\beta)$ is concave in β hence if we find a local maximum it is a global maximum.

We have for $(k, l) \in (1, \cdots, p)^2$:

$$
\begin{aligned}
\frac{\partial \ell}{\partial \beta_k \partial \beta_l}(\beta) =& \frac{\partial}{\partial \beta_k} \sum_{i=1}^{n} x_{il}\left(y_i - \frac{\exp(x_i^T \beta)}{1 + \exp(x_i^T \beta)}\right) \\
=& - \sum_{i=1}^{n} x_{il} x_{ik} \frac{\exp(x_i^T \beta)}{(1 + \exp(x_i^T \beta))^2} \\
=& - \sum_{i=1}^{n} x_{ik} p_\beta(x_i)(1 - p_\beta(x_i)) x_{il}
\end{aligned}
$$

Estimation

MLE existence

We obtain that in matrix form:

$$H(\beta) = \nabla^2 \ell(Y, \beta) = -X^T W_\beta X$$

where:

$$W_\beta = \begin{pmatrix} p_\beta(x_1)(1 - p_\beta(x_1)) & \cdots & \cdots \\ \vdots & \ddots & \vdots \\ \cdots & \cdots & p_\beta(x_n)(1 - p_\beta(x_n)) \end{pmatrix}$$

We have $p_\beta(x_i)(1 - p_\beta(x_i)) \geq 0$ hence $W(\beta)$ is semi-definite negative and since $rank(X) = p$, $H(\beta)$ is concave.

Estimation

MLE existence

It is shown in (Albert & Aanderson, 1984) that if additionally there is no complete separation in the training set:

Maximum likelihood estimates in logistic regression

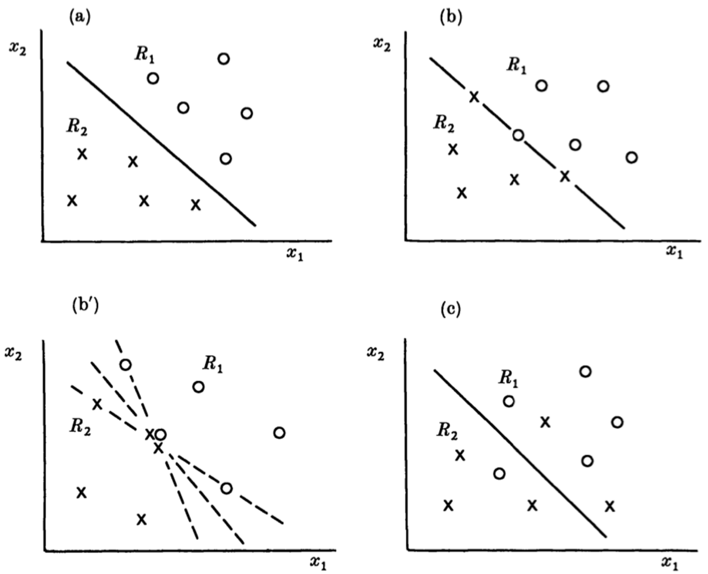


Fig. 1. Possible configurations of sample points in the case of two variables, x_1 and x_2, and two groups, H_1, shown by circles, and H_2, shown by crosses. Regions R_1 and R_2 define corresponding allocation rule. (a) Complete separation. (b) Quasicomplete separation $\rho(X^q) = 2$. (b′) Quasi-complete separation $\rho(X^q) = 1$; at the point of intersection of the lines, there are three observations, one from H_1, and two from H_2. (c) Overlap.

then the MLE exists and is unique, we will denote it $\hat{\beta}$.

Estimation

The Newton-Raphson method

In practice a numerical/iterative method such as the Newton-Raphson method is used to solve the equation:

$$\boldsymbol{S}(\beta) = \nabla \ell(\boldsymbol{Y}, \beta) = \boldsymbol{X}^T(\boldsymbol{Y} - \boldsymbol{P}_\beta) = \boldsymbol{0}$$

Using Taylor expansion of Score $\boldsymbol{S}(\beta)$ around an initial guess $\beta^{(0)}$ of $\hat{\beta}$:

$$\boldsymbol{S}(\hat{\beta}) \approx \boldsymbol{S}(\beta^{(0)}) + \boldsymbol{H}(\beta^{(0)})(\hat{\beta} - \beta^{(0)})$$

giving a first estimate of $\hat{\beta}$:

$$\beta^{(1)} = \beta^{(0)} - \boldsymbol{H}^{-1}(\beta^{(0)})\boldsymbol{S}(\beta^{(0)})$$

Then until convergence, the Newton-Raphson method iterates:

$$\beta^{(k+1)} = \beta^{(k)} - \boldsymbol{H}^{-1}(\beta^{(k)})\boldsymbol{S}(\beta^{(k)})$$

Estimation

The Newton-Raphson method

We show below a naive implementation of Newton-Raphson method to estimate β:

▾ Code

```
# We put the data frame in matrix form
# also adding an intercept
X <- cbind(rep(1, nrow(default_data)),
                          as.matrix(default_data %>% select(balance, income)))
colnames(X) <-  c("(Intercept)", "balance", "income")
n <- nrow(X)

# We extract the output as vector
Y <- default_data %>% mutate(default = if_else(default=='Yes', 1, 0)) %>% pull(default)

# We set an initial guess for beta and criterion for stopping
beta <- c(0.01, 0.0, 0.0)
nb_iter <- 25
tol <- 1e-4

lr_solve <- function(X, Y, beta, nb_iter, tol){
    for(i in 1:nb_iter){
        # first compute p_beta(X)
        p_beta <- exp(X %*% beta) / (1 + exp(X %*% beta))
        # then the Score
        Score_beta <- t(X) %*% (Y-p_beta)
        # and the Hessian
        W_beta <- matrix(0, n, n)
        diag(W_beta) <- p_beta*(1-p_beta)
        Hessian_beta <- -t(X) %*% W_beta %*% X
        # we update beta
        new_beta <- beta - solve(Hessian_beta) %*% Score_beta
        # we check for convergence
```

Estimation

The Newton-Raphson method

We verify that the R `glm()` function and our algorithm give close values for coefficients β:

- R `glm()`:

```
(Intercept)     balance      income
 -11.540468    0.005647    0.000021
```

- Newton-Raphson:

```
     (Intercept)  balance  income
[1,]   -11.53791 0.005646 2.1e-05
```

Estimation

Machine learning "point of view"

We can rewrite the log-likelihood equation stated before:

$$\begin{aligned}\ell(Y,\beta) = \log L(Y,\beta) &= \sum_{i=1}^{n} \left(y_i \log(p_\beta(x_i)) + (1-y_i)\log(1-p_\beta(x_i))\right)\\ &= -\sum_{i=1}^{n} \ell_{logistic}\left(p_\beta(x_i), y_i\right)\\ &= -n\hat{\mathrm{R}}(p_\beta)\end{aligned}$$

where $\ell_{logistic} : \{0,1\} \times \{0,1\} \to \mathbb{R}^+$:

$$\ell_{logistic}(y,z) = -y\log(z) - (1-y)\log(1-z) = \begin{cases} -\log(z) & \text{if } y = 1\\ -\log(1-z) & \text{if } y = 0\end{cases}$$

and $\hat{\mathrm{R}}(p_\beta)$ is the empirical risk on the training set.

Estimating β by maximizing the log-likelihood is equivalent to minimizing with respect to β the empirical risk of p_β for the logistic loss.

Inference

Inference

Interpretation - the R `glm()` output

▾ Code

```
summary(glm_default)
```

```
Call:
glm(formula = default ~ ., family = "binomial", data = default_data)

Coefficients:
              Estimate Std. Error z value Pr(>|z|)    
(Intercept) -1.087e+01  4.923e-01 -22.080  < 2e-16 ***
studentYes  -6.468e-01  2.363e-01  -2.738  0.00619 ** 
balance      5.737e-03  2.319e-04  24.738  < 2e-16 ***
income       3.033e-06  8.203e-06   0.370  0.71152    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2920.6  on 9999  degrees of freedom
Residual deviance: 1571.5  on 9996  degrees of freedom
AIC: 1579.5

Number of Fisher Scoring iterations: 8
```

Based on this output, the fitted model is (we have re-scaled balance and income for better readability):

$$\log\left(\frac{p_\beta(x_i)}{1-p_\beta(x_i)}\right) = -1.08 - 0.65(\mathbb{1}_{\text{student}_i=\text{Yes}}) + 5.74(\frac{\text{balance}_i}{1000}) + 0.03(\frac{\text{income}_i}{10000})$$

Inference

Interpretation - logits

$$\log\left(\frac{p_\beta(x_i)}{1-p_\beta(x_i)}\right) = -1.08 - 0.65(\mathbb{1}_{\text{student}_i=\text{Yes}}) + 5.74(\frac{\text{balance}_i}{1000}) + 0.03(\frac{\text{income}_i}{10000})$$

Coefficients interpretation

We cannot interpret the coefficients in the same manner as we interpret coefficients from a linear model, as the outcome is now expressed in "logits":

- The predicted logit (or as we will see later log-odds) of defaulting for non-students with zero balance and income are -1.08.
- Each one-unit difference in $\frac{\text{balance}}{1000}$ is associated with a difference of 5.74 in the predicted logit of defaulting.
- Each one-unit difference in $\frac{\text{income}}{10000}$ is associated with a difference of 0.03 in the predicted logit of defaulting.

Inference

Interpretation - odds

We remind the following relationship:

$$\text{logit}(p_\beta(x)) = \log(\frac{p_\beta(x)}{1 - p_\beta(x)}) = x^T\beta$$

The ratio on which we take the logarithm is called odds:

$$odd_\beta(x) = \frac{p_\beta(x)}{1 - p_\beta(x)} = exp(x^T\beta)$$

It represents the chance an event occurs ($p_\beta(x)$) versus the chance that same event does not occur ($1 - p_\beta(x)$).

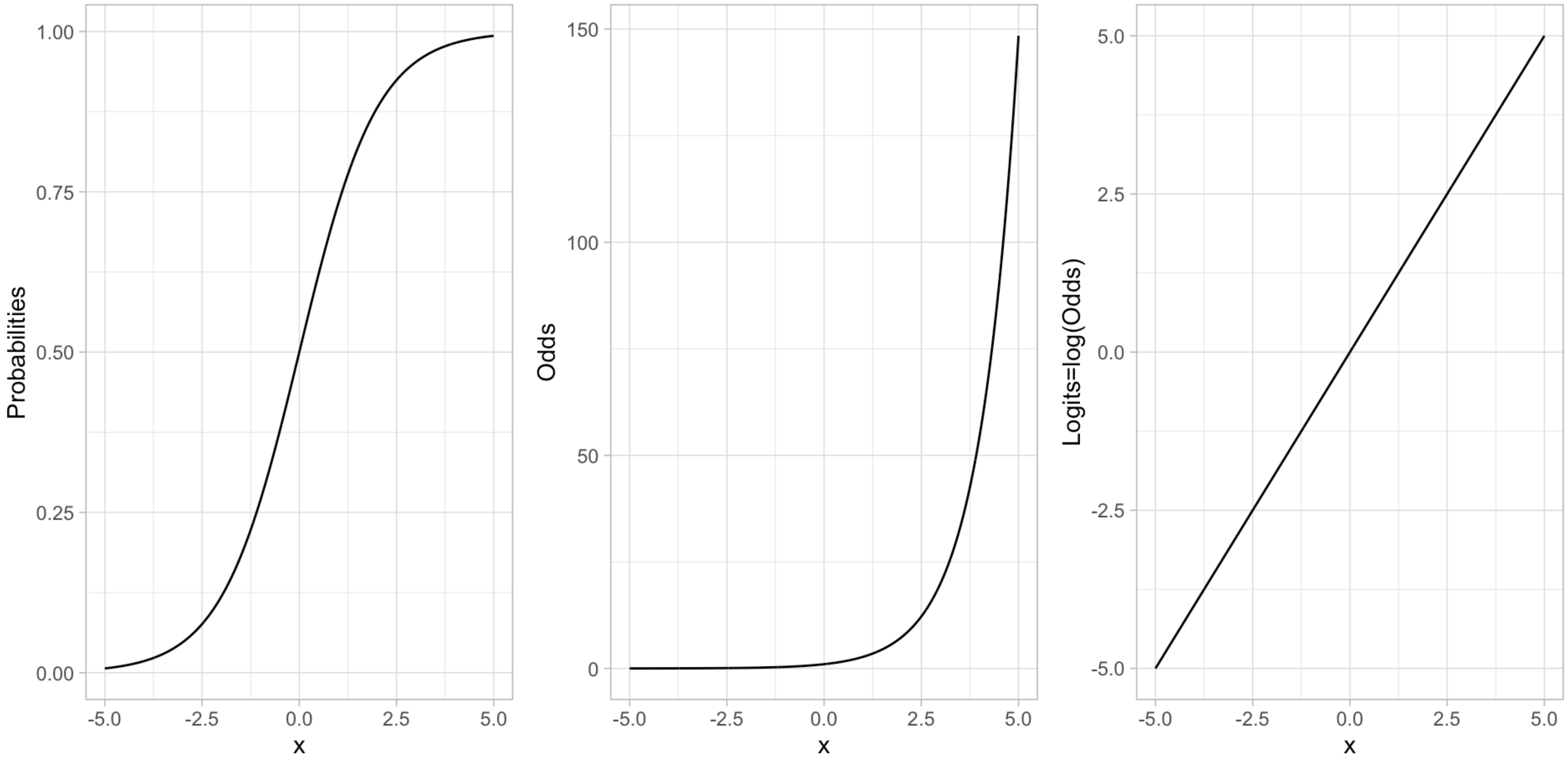
We can also rewrite:

$$p_\beta(x) = \frac{odds_\beta(x)}{1 + odds_\beta(x)}$$

Inference

Interpretation - odds

To set these ideas, for a variable x in $[-5, 5]$, we plot the logistic curve (i.e. the probabilities) together with the odds and the logits (ie log(odds)):



Inference

Interpretation - odds ratio

For two observations $\boldsymbol{x}$ and $\tilde{\boldsymbol{x}}$ we define odds ratio as:

$$OR(x, \tilde{x}) = \frac{odds(x)}{odds(\tilde{x})}$$

Odds ratio are used to compare probabilities between two observations:

- $OR(x, \tilde{x}) = 1 \Leftrightarrow p(x) = p(\tilde{x})$
- $OR > 1 \Leftrightarrow p(x) > p(\tilde{x})$
- $OR < 1 \Leftrightarrow p(x) < p(\tilde{x})$

They are also used to measure the impact of a predictor:

$$OR(x, \tilde{x}) = \exp(\beta_1(x_1 - \tilde{x_1})) \cdots exp(\beta_p(x_p - \tilde{x_p}))$$

Choosing $(\boldsymbol{x}, \tilde{\boldsymbol{x}})$ differing by only one predictor $\boldsymbol{x}_j$:

$$OR(x, \tilde{x}) = \exp(\beta_j(x_j - \tilde{x_j}))$$

Inference

Interpretation - odds ratio

$exp(\beta_j)$ is the odds ratio associated with a one-unit increase in the x_j.

Tip

Interpret the coefficients in terms of odds:

- The coefficient of balance / 1000 is 5.737. Hence an increase of balance by 1000 points increases odds for default by a factor of exp(5.737)=310.
- The coefficient of income / 10000 is 0.03. Hence an increase of income by 10000 points increases odds for default by a factor of exp(0.03)=1.03.
- The coefficient of "being a student" is -0.647. Hence being a student decreases odds for default by a factor of exp(-0.647)=0.52.

More on odds ratio interpretation can be found here.

Inference

Asymptotic properties of MLE

Under certain assumptions (see for example Gourieroux (1981) or Fahrmeir (1986)), the Maximum Likelihood Estimator has the following asymptotic properties:

$$\hat{\beta} \xrightarrow{p} \beta, \textbf{ as n} \to \infty$$

and

$$\sqrt{n}(\hat{\beta} - \beta) \xrightarrow{\mathcal{L}} \mathcal{N}(0, \mathcal{I}(\beta)^{-1}), \textbf{ as n} \to \infty$$

where:

$$\mathcal{I}(\beta) = -\mathbb{E}[\nabla^2 \ell(Y, \beta)] = -\frac{1}{n}\nabla^2 \ell(Y, \beta) = \frac{1}{n} X^T W_\beta X$$

where $\mathcal{I}(\beta)$ is the Fisher information matrix. We recognize the Hessian matrix we have seen before.

Inference

Asymptotic properties of MLE

The asymptotic property rewrites:

$$(\hat{\beta} - \beta)^T n\mathcal{I}(\beta)(\hat{\beta} - \beta) \xrightarrow{\mathcal{L}} \chi^2_p$$

As $\mathcal{I}(\beta)$ is unknown we use instead $\mathcal{I}(\hat{\beta}) = \frac{1}{n} X^T W_{\hat{\beta}} X$. Since $\hat{\beta} \xrightarrow{p} \beta$ and p_β continuous in β it can be shown that:

$$(\hat{\beta} - \beta)^T X^T W_{\hat{\beta}} X(\hat{\beta} - \beta) \xrightarrow{\mathcal{L}} \chi^2_p$$

Or equivalently:

$$\hat{\beta} - \beta \xrightarrow{\mathcal{L}} \mathcal{N}(0, (X^T W_{\hat{\beta}} X)^{-1})$$

Inference

Wald statistics

Using the preceding asymptotic properties we can derive confidence interval and tests for the coefficients β_j, $j = 1, \cdots, p$ of the model:

$$\frac{\hat{\beta}_j - \beta_j}{\hat{\sigma}_j} \xrightarrow{\mathcal{L}} \mathcal{N}(0,1)$$

where $\hat{\sigma}_j^2 = s.e.(\hat{\beta}_j)^2$ denotes the $j-th$ term of $(X^T W_{\hat{\beta}} X)^{-1}$ diagonal.

The typical formula for a $1-\alpha$ confidence interval is:

$$\hat{\beta}_j \pm z_{1-\alpha/2} \hat{\sigma}_j$$

where $z_{1-\alpha/2}$ is the $(1-\alpha/2)$ quantile of the standard normal distribution.

Inference

Wald statistics

Going further, the asymptotic properties of MLE also allow to test the “statistical significance” of each coefficient in the model, the Wald test.

Denoting: $\mathbf{H}_0$: $\beta_j = 0$ and $\mathbf{H}_1$: $\beta_j \neq 0$ we have under $\mathbf{H}_0$:

$$\frac{\hat{\beta}_j}{\hat{\sigma}_j} \xrightarrow{\mathcal{L}} \mathcal{N}(0, 1)$$

We will reject $\mathbf{H}_0$ at level α if the absolute of the observed value $\frac{\hat{\beta}_j}{\hat{\sigma}_j}$ (denoted in `glm` output as `z value`) is above the $(1 - \alpha/2)$ quantile of the standard normal distribution.

Inference

Wald statistics

Code

```r
glm_bal_inc <- glm(default ~ balance + income,
                   data = default_data,
                   family = "binomial")
summary(glm_bal_inc)
```

```
Call:
glm(formula = default ~ balance + income, family = "binomial",
    data = default_data)

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.154e+01  4.348e-01 -26.545  < 2e-16 ***
balance      5.647e-03  2.274e-04  24.836  < 2e-16 ***
income       2.081e-05  4.985e-06   4.174 2.99e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2920.6  on 9999  degrees of freedom
Residual deviance: 1579.0  on 9997  degrees of freedom
AIC: 1585

Number of Fisher Scoring iterations: 8
```

Inference

Wald statistics

We reject $\mathbf{H}_0$ at level α when $p = \mathbf{P}(|z| > |\frac{\hat{\beta}_j}{\hat{\sigma}_j}|) < \alpha$.

p is called the p-value.

The output of `glm` in `R` shows:

- $\hat{\beta}_j$ as `Estimate`,
- $\hat{\sigma}_j$ as `Std. Error`,
- the observed test statistic $\frac{\hat{\beta}_j}{\hat{\sigma}_j}$ as `z value`,
- and the p-value as `Pr(>|z|)`

Inference

Wald statistics

We obtain the `z value` $\frac{\hat{\beta}_j}{\hat{\sigma}_j}$ using estimate and its standard deviation:

▾ Code

```
z <- glm_bal_inc$coefficients[3] / (summary(glm_bal_inc))$coefficients[3,2]
z
```

```
  income
4.174178
```

and then the p-value:

▾ Code

```
(1-pnorm(abs(z)))*2
```

```
      income
2.990638e-05
```

Inference

Wald statistics

Using the hessian matrix obtained before as a side product of the Newton-Raphson algorithm, we retrieve comparable values with `glm` outputs for $\hat{\sigma}_j$:

▾ Code

```
std_errors <- sqrt(diag(solve(-as.matrix(sol$hessian))))
std_errors
```

```
 (Intercept)      balance       income 
4.346349e-01 2.273110e-04 4.984579e-06 
```

the `z values` and then the p-values:

▾ Code

```
z <- sol$beta / std_errors
t(z)
```

```
     (Intercept)  balance   income
[1,]   -26.54622 24.83714 4.173579
```

▾ Code

```
t((1-pnorm(abs(z)))*2)
```

```
     (Intercept) balance       income
[1,]           0       0 2.998516e-05
```

Inference

Wald statistics - confidence intervals

The R command to get confidence interval of estimators based on Wald statistic is the following (by default $\alpha = 5\%$)

Code

```
confint.default(glm_bal_inc)
```

```
                    2.5 %        97.5 %
(Intercept) -1.239258e+01 -1.068836e+01
balance      5.201460e-03  6.092746e-03
income       1.103823e-05  3.057972e-05
```

We can retrieve it manually using coefficient estimate and standard deviation:

Code

```
output_bal_inc = summary(glm_bal_inc)$coefficients
bal_std_estimate <- output_bal_inc[2,1]
bal_std_error <- output_bal_inc[2,2]

# upper bound for beta(balance) at 5%
upper <- bal_std_estimate + 1.96 * bal_std_error

# lower bound for beta(balance) at 5%
lower <- bal_std_estimate - 1.96 * bal_std_error
(bal_confint <- c(lower, upper))
```

```
[1] 0.005201452 0.006092754
```

Inference

Confidence intervals

The following R command provides confidence interval of estimators using a more advanced profile likelihood method:

Code

```
confint(glm_bal_inc)
```

```
                    2.5 %        97.5 %
(Intercept) -1.241910e+01 -1.071361e+01
balance      5.214030e-03  6.105971e-03
income       1.105359e-05  3.060844e-05
```

More details on profile likelihood method can be found here.

Inference

Wald statistics - tests on model coefficients

Based on the same idea, it is possible to test for the nullity of a subset of the model coefficients.

Denoting: $\mathbf{H_0}$: $\beta_1 = \cdots = \beta_q = 0$, $\mathbf{H_1}$: $\exists j \in \{1, \cdots, q\} \mid \beta_j \neq 0$, $\hat{\beta} = (\hat{\beta}_1, \cdots, \hat{\beta}_p)$ the MLE and $\hat{\beta}_{1:q} = (\hat{\beta}_1, \cdots, \hat{\beta}_q)$ the vector of first q parameters.

We have under $\mathbf{H_0}$:

$$\hat{\beta}^T_{1:q}(X^T W_{\hat{\beta}} X)^{-1}_{1:q}\hat{\beta}_{1:q} \xrightarrow{\mathcal{L}} \chi^2_q$$

where $(X^T W_{\hat{\beta}} X)^{-1}_{1:q}$ is the $q \times q$ upper left block matrix extracted from the inverse of hessian.

We will reject $\mathbf{H_0}$ at level α if the observed value $\hat{\beta}^T_{1:q}(X^T W_{\hat{\beta}} X)^{-1}_{1:q}\hat{\beta}_{1:q}$ is above the $1 - \alpha$ quantile of the χ^2_q distribution.

Inference

Inference - tests on model coefficients

We show below the Wald tests for each coefficient in the model using `summary`:

Code

```
summary(glm_default)
```

```
Call:
glm(formula = default ~ ., family = "binomial", data = default_data)

Coefficients:
              Estimate Std. Error z value Pr(>|z|)    
(Intercept) -1.087e+01  4.923e-01 -22.080  < 2e-16 ***
studentYes  -6.468e-01  2.363e-01  -2.738  0.00619 ** 
balance      5.737e-03  2.319e-04  24.738  < 2e-16 ***
income       3.033e-06  8.203e-06   0.370  0.71152    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2920.6  on 9999  degrees of freedom
Residual deviance: 1571.5  on 9996  degrees of freedom
AIC: 1579.5

Number of Fisher Scoring iterations: 8
```

Inference

Wald statistics - tests on model coefficients

These tests can be also performed in `R` using `car::Anova` or `aod::wald.test` routines. In particular when categorical variables have more than two levels these functions allow to test each variables as a whole (vs coefficient by coefficient when using `summary`)

Code

```
car::Anova(glm_default, type=3, test.statistic= "Wald")
```

```
Analysis of Deviance Table (Type III tests)

Response: default
            Df    Chisq Pr(>Chisq)    
(Intercept)  1 487.5303  < 2.2e-16 ***
student      1   7.4947   0.006188 ** 
balance      1 611.9470  < 2.2e-16 ***
income       1   0.1368   0.711520    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Inference

Wald statistics - tests on model coefficients

Code

```
# Testing the income coefficient (Terms = 4)
aod::wald.test(b = coef(glm_default), Sigma = vcov(glm_default), Terms = 4)
```

```
Wald test:
----------

Chi-squared test:
X2 = 0.14, df = 1, P(> X2) = 0.71
```

Code

```
# Testing the income coefficient (Terms = 4)
aod::wald.test(b = coef(glm_default), Sigma = vcov(glm_default), Terms = 2)
```

```
Wald test:
----------

Chi-squared test:
X2 = 7.5, df = 1, P(> X2) = 0.0062
```

Inference

Wald statistics - tests on model coefficients

We can retrieve these outputs manually:

▾ Code

```r
sum_default <- summary(glm_default)
beta_income <- sum_default$coefficients[4,1]
stdev_income <- sum_default$coefficients[4,2]

wald <- beta_income ^ 2 / stdev_income ^ 2

1-pchisq(wald, df = 1)
```

```
[1] 0.7115203
```

▾ Code

```r
z_val <- sum_default$coefficients[4,3]

2*(1-pnorm(abs(z_val)))
```

```
[1] 0.7115203
```

With all routines, the p-value for the income coefficient is 0.71 validating the null hypothesis.

Inference

Wald statistics - tests on model coefficients

Using `Terms` or `L` parameters in `aod::wald.test` it is also feasible to test the null hypothesis for a subsets of parameters:

Code

```
# Testing the income coefficient (Terms = 4)
aod::wald.test(b = coef(glm_default), Sigma = vcov(glm_default), Terms = 1:3)
```

```
Wald test:
----------

Chi-squared test:
X2 = 698.3, df = 3, P(> X2) = 0.0
```

The null hypothesis is rejected for the model with balance and student.

Inference

Likelihood ratio tests

It is possible to test for the nullity of a subset of the model coefficients using Likelihood Ratio statistics.

Denoting: $\mathbf{H_0}$: $\beta_1 = \cdots = \beta_q = 0$, $\mathbf{H_1}$: $\exists j \in \{1, \cdots, q\} \mid \beta_j \neq 0$, and $\hat{\beta} = (\hat{\beta}_1, \cdots, \hat{\beta}_p)$ the MLE, we have under $\mathbf{H_0}$:

$$-2\left(\ell_{\mathrm{H}_0}(Y, \hat{\beta}_{\mathrm{H}_0}) - \ell(Y, \hat{\beta})\right) \overset{\mathcal{L}}{\to} \chi^2_q$$

where $\ell_{\mathrm{H}_0}(Y, \hat{\beta}_{\mathrm{H}_0})$ is the log-likelihood of:

$$\mathrm{logit}(p_\beta(X)) = x_{q+1}\beta_{q+1} + \cdots + x_n\beta_n$$

Inference

Likelihood ratio tests

Consider two models, a larger model with l parameters and likelihood L_L and a smaller model with s parameters and likelihood L_S, where the smaller model represents a subset of the larger model. Typically the smaller model is equivalent to the large model where we have imposed:

$$\mathrm{H}_0\text{: } \beta_j = \ldots = \beta_{j+r} = 0$$

Likelihood Ratio tests on variables may be performed in R using `car::Anova`:

Code

```
car::Anova(glm_default, type=3, test.statistic= "LR")
```

```
Analysis of Deviance Table (Type III tests)

Response: default
        LR Chisq Df Pr(>Chisq)
student     7.42  1   0.006445 **
balance  1335.95  1  < 2.2e-16 ***
income      0.14  1   0.711514
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Inference

Likelihood ratio tests

Using base `R` `anova` it is also possible to test subsets of variables and in particular individual variables within the "full" model, we have to fit `glm` for each sub model:

▾ Code

```
glm_wo_student <- glm(default ~ balance + income,
                 data = default_data,
                 family = "binomial")
glm_wo_balance <- glm(default ~ student + income,
                 data = default_data,
                 family = "binomial")
glm_wo_income <- glm(default ~ student + balance,
                 data = default_data,
                 family = "binomial")
```

Inference

Likelihood ratio tests

Testing the model without student predictor:

Code

```
anova(glm_wo_student, glm_default, test = "LRT")
```

```
Analysis of Deviance Table

Model 1: default ~ balance + income
Model 2: default ~ student + balance + income
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1      9997     1579.0
2      9996     1571.5  1   7.4214 0.006445 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We can retrieve this result manually:

Code

```
LRT <- 2 * (logLik(glm_default)-logLik(glm_wo_student))
1 - pchisq(LRT, df = 1)
```

```
'log Lik.' 0.006445112 (df=4)
```

Inference

Likelihood ratio tests

Testing the other sub models:

▾ Code

```
anova(glm_wo_balance, glm_default, test = "LRT")
```

```
Analysis of Deviance Table

Model 1: default ~ student + income
Model 2: default ~ student + balance + income
  Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
1      9997     2907.5
2      9996     1571.5  1     1336 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

▾ Code

```
anova(glm_wo_income, glm_default, test = "LRT")
```

```
Analysis of Deviance Table

Model 1: default ~ student + balance
Model 2: default ~ student + balance + income
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1      9997     1571.7
2      9996     1571.5  1  0.13677   0.7115
```

The deviance defined as $D = -2\ell$ is often reported by statistical software in place of log-likelihood. A large likelihood corresponding to a small deviance.

Inference

Likelihood ratio tests

A better and full coverage of tests in the context of Logistic Regression can be found here or in (Hosmer et al., 2013). See also here for a data analysis using **R** and see the question here for a very detailed description of the outputs of `glm()` (in particular this answer).

Goodness of Fit

Goodness of Fit

Hosmer & Lemeshow test

Although it is generally not recommended by practitioners and theoreticians, the Hosmer & Lemeshow test (see here, here or here) allows to quickly assess the “goodness of fit” of a Logistic Regression.

But more than the test in itself, the underlying motivation is interesting: the Logistic Regression model provides an estimate of the probability of an outcome (success/failure, here the default is success or 1). The estimated probability of this outcome should be close to the true observed probability.

The Hosmer & Lemeshow test assess if observed event rates match expected event rates in subgroups of “similar” observations. Models for which expected and observed event rates agree on these subgroups are considered well calibrated.

Goodness of Fit

Hosmer & Lemeshow test

A first step of the test is to order the predicted probabilities of the outcome and divide it into Q groups (usually using deciles, Q=10). Then the average predicted probability for each group is computed and compared to the observed probability.

The Hosmer & Lemeshow test statistic $\boldsymbol{H}$ is compared to a χ^2_{Q-2} distribution:

$$H = \sum_{q=1}^{Q} \frac{(o_q - m_q\mu_q)^2}{m_q\mu_q(1-\mu_q)}$$

where:

- $\boldsymbol{o_q}$ denotes the number of success $(\boldsymbol{Y=1})$ observed in group $\boldsymbol{q}$,
- $\boldsymbol{\mu_q}$ denotes the mean of $\boldsymbol{p_{\hat{\beta}}(x_i)}$ in group $\boldsymbol{q}$,
- $\boldsymbol{m_q}$ denotes the number of observations in group $\boldsymbol{q}$, so that $\boldsymbol{m_q\mu_q}$ is the expected number of success in group $\boldsymbol{q}$.

The null hypothesis is that observed/expected outcomes are close along all subgroups.

Goodness of Fit

Hosmer & Lemeshow test

▾ Code

```
library(glmtoolbox)
hltest(glm_default)
```

```
   The Hosmer-Lemeshow goodness-of-fit test

 Group Size Observed     Expected
     1 1000        0   0.02653992
     2 1000        0   0.10737240
     3 1000        0   0.29143249
     4 1000        1   0.67265778
     5 1000        2   1.39515666
     6 1000        1   2.87108745
     7 1000        7   5.98948667
     8 1000       16  13.74542953
     9 1000       45  39.52811751
    10 1000      261 268.37271994

         Statistic =  3.68229
degrees of freedom =  8
           p-value =  0.88459
```

Here the p-value for a chi-squared statistic of $H = 3.68$ with $df = Q - 2 = 8$ is $p = 0.885$ which is well above the usual levels (eg 0.05), so that the null hypothesis is accepted (ie observed and expected probabilities agree across the Q subgroups), goodness of fit is acceptable.

However the Hosmer & Lemeshow test is dependent on the choice of Q and the binning performed on probabilities and is sometimes considered unreliable.

Goodness of Fit

Calibration plots

Nonetheless it is usual to assess or diagnose the good calibration of a model probabilities using Calibration Plots or Probability Calibration Curves (see here for a recent R package from the tidyverse/tidymodel ecosystem and here for a scikit-learn version): they are used to visualize if predictions are consistent with the observed event rates (be it on the training set or a testing set, which is better).

Goodness of Fit

Calibration plots

For example considering the `default` data set we have:

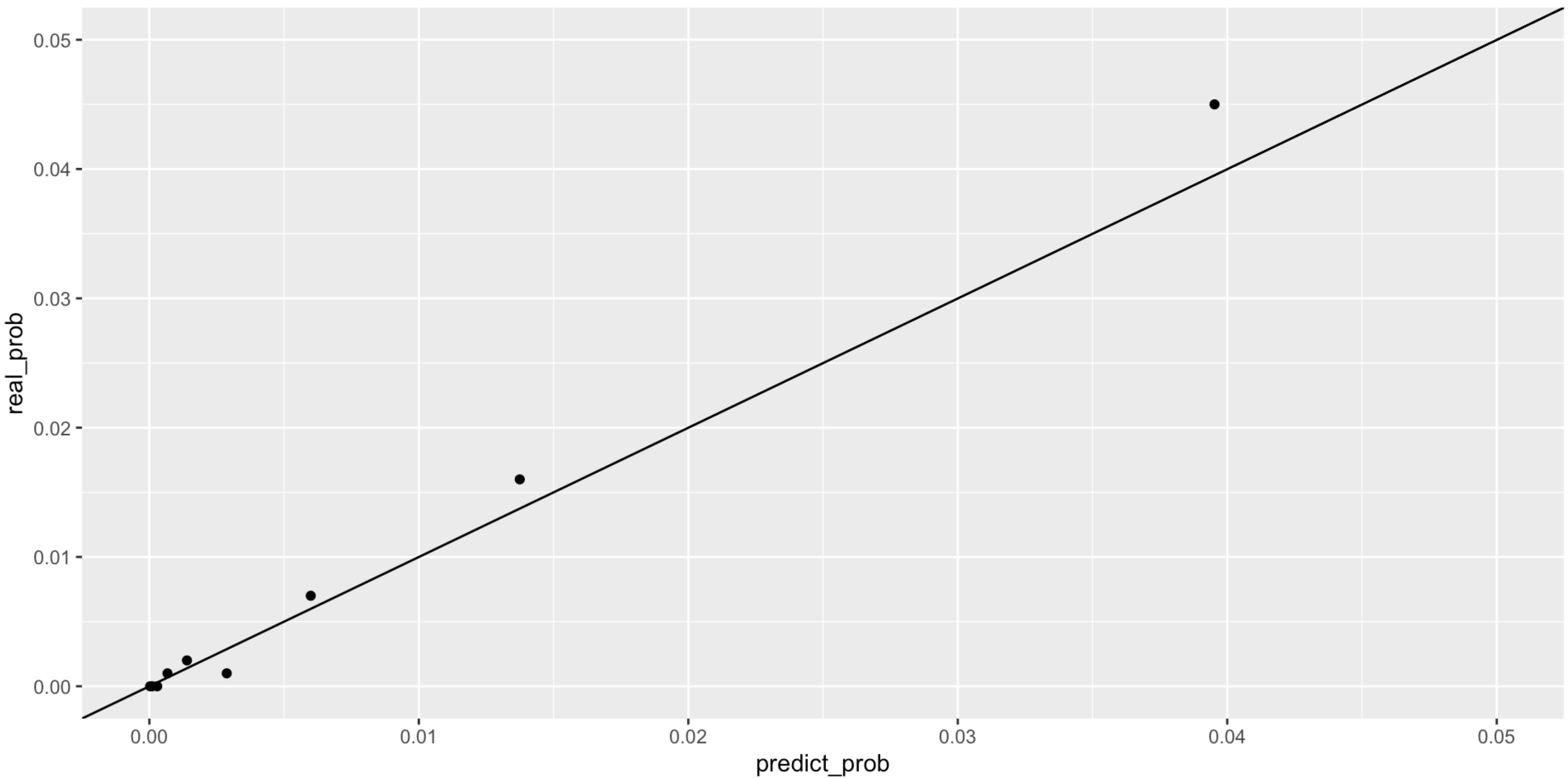
▾ Code

```r
check_default_prob <- as_tibble(cbind(fitted=glm_default$fitted.values,
                                       Y = default_data %>%
                                         mutate(default = if_else(default == "Yes", 1, 0)) %>%
                                         pull(default)))
(calibration_data <- check_default_prob %>%
  mutate(bins_prob = cut(fitted, breaks = quantile(fitted,seq(0,1,0.10)), include.lowest = TRUE)) %>%
  group_by(bins_prob) %>%
  summarize(n = n(),
            def = sum(Y),
            no_def = n - def,
            predict_prob = mean(fitted),
            real_prob = def/n,
            forecast_acc = def / sum(check_default_prob$Y)))
```

```
# A tibble: 10 × 7
   bins_prob                n   def no_def predict_prob real_prob forecast_acc
   <fct>                <int> <dbl>  <dbl>        <dbl>     <dbl>        <dbl>
 1 [1.03e-05,5.14e-05]   1000     0   1000    0.0000265     0            0
 2 (5.14e-05,0.000176]   1000     0   1000    0.000107      0            0
 3 (0.000176,0.000443]   1000     0   1000    0.000291      0            0
 4 (0.000443,0.000945]   1000     1    999    0.000673      0.001        0.00300
 5 (0.000945,0.00197]    1000     2    998    0.00140       0.002        0.00601
 6 (0.00197,0.00402]     1000     1    999    0.00287       0.001        0.00300
 7 (0.00402,0.0088]      1000     7    993    0.00599       0.007        0.0210
 8 (0.0088,0.021]        1000    16    984    0.0137        0.016        0.0480
 9 (0.021,0.0709]        1000    45    955    0.0395        0.045        0.135
10 (0.0709,0.978]        1000   261    739    0.268         0.261        0.784
```

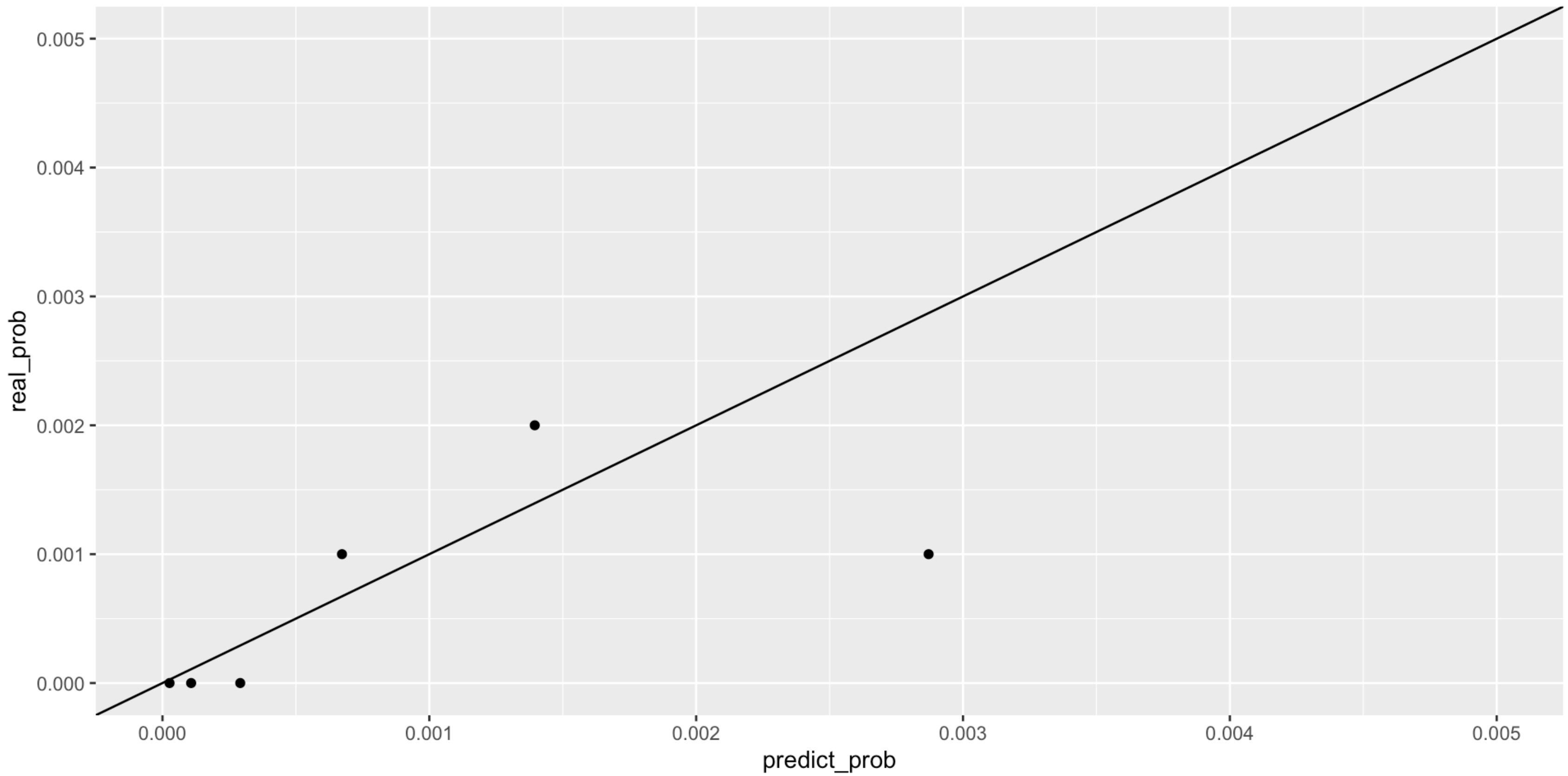
Goodness of Fit

Calibration plots



Goodness of Fit

Calibration plots



The model seems to slightly overestimate/underestimates depending on considered deciles, with no clear pattern of bias.

References

Albert, A., & Aanderson, J. A. (1984). On the existence of maximum likelihood estimates in logistic regression models. *Biometrika*, *71*(1), 1–10. https://doi.org/10.1093/biomet/71.1.1

Fahrmeir, &. K., L. (1986). Asymptotic inference in discrete response models. *Statistische Hefte*, *27*(1), 179–205. https://doi.org/10.1007/bf02932567

Gourieroux, &. M., C. (1981). Asymptotic properties of the maximum likelihood estimator in dichotomous logit models. *Journal of Econometrics*, *17*(1), 83–97. https://doi.org/10.1016/0304-4076(81)90060-9

Hosmer, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). Applied logistic regression. In *Wiley Series in Probability and Statistics*. https://doi.org/10.1002/9781118548387

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2021). *An introduction to statistical learning: With applications in r*. Springer US. https://doi.org/10.1007/978-1-0716-1418