Lecture 6. Binary regression. Applied Statistics in R

Elena Parilina

Master's Program
Game Theory and Operations Research

Saint Petersburg State University

2020

Agenda

Binary logistic regression

Binary logistic regression

$y \in \{0, 1\}$: dependent variable,
$x_1, \ldots, x_k$: predictor (explanatory) variables.

$x = (1, x_1, \ldots, x_k)^\mathsf{T}$

Sample $(1, x_{i1}, \ldots, x_{ik}, y_i)$, $i = 1, \ldots, n$. The model

$$\mathsf{P}\{y_i = 1\} = F(\beta^T x_i).$$

Let $y_i^* = \beta^T x_i + \varepsilon_i$ where $\varepsilon_1, \ldots, \varepsilon_n$: i.i.d. with $E\varepsilon_i = 0$ and $\operatorname{var} \varepsilon_i = \sigma^2$.

Binary logistic regression

$y \in \{0,1\}$: dependent variable,
$x_1, \dots, x_k$: predictor (explanatory) variables.

$x = (1, x_1, \dots, x_k)^\mathsf{T}$

Sample $(1, x_{i1}, \dots, x_{ik}, y_i)$, $i = 1, \dots, n$. The model

$$\mathsf{P}\{y_i = 1\} = F(\beta^T x_i).$$

Let $y_i^* = \beta^T x_i + \varepsilon_i$ where $\varepsilon_1, \dots, \varepsilon_n$: i.i.d. with $E\varepsilon_i = 0$ and $\operatorname{var}\varepsilon_i = \sigma^2$.

$$y_i = \begin{cases} 1, & y_i^* \geqslant 0, \\ 0, & y_i^* < 0. \end{cases}$$

Binary logistic regression

$y \in \{0,1\}$: dependent variable,
$x_1, \ldots, x_k$: predictor (explanatory) variables.

$x = (1, x_1, \ldots, x_k)^\mathsf{T}$

Sample $(1, x_{i1}, \ldots, x_{ik}, y_i)$, $i = 1, \ldots, n$. The model

$$\mathsf{P}\{y_i = 1\} = F(\beta^T x_i).$$

Let $y_i^* = \beta^T x_i + \varepsilon_i$ where $\varepsilon_1, \ldots, \varepsilon_n$: i.i.d. with $E\varepsilon_i = 0$ and $\operatorname{var} \varepsilon_i = \sigma^2$.

$$y_i = \begin{cases} 1, & y_i^* \geqslant 0, \\ 0, & y_i^* < 0. \end{cases}$$

$$\mathsf{P}\{y_i = 1\} = \mathsf{P}\{y_i^* \geqslant 0\} = \mathsf{P}\{\beta^T x_i + \varepsilon_i \geqslant 0\} = \mathsf{P}\{\varepsilon_i \geqslant -\beta^T x_i\} =$$
$$= 1 - F\left(-\frac{\beta^T x_i}{\sigma}\right) = F\left(\frac{\beta^T x_i}{\sigma}\right),$$

assuming $F(x)$ is a continuous CDF of ε_i/σ and $F(-x) = 1 - F(x)$.

Two models (depending on F):

- Probit model $\Phi(u) = \frac{1}{\sqrt{2\pi}} \int\limits_{-\infty}^{u} e^{-\frac{z^2}{2}} dz.$

Two models (depending on F):

- Probit model $\Phi(u) = \frac{1}{\sqrt{2\pi}} \int\limits_{-\infty}^{u} e^{-\frac{z^2}{2}} dz.$
- Logit model $\Lambda(u) = \frac{e^u}{1 + e^u}.$

Two models (depending on F):

- Probit model $\Phi(u) = \dfrac{1}{\sqrt{2\pi}} \displaystyle\int_{-\infty}^{u} e^{-\frac{z^2}{2}} dz.$
- Logit model $\Lambda(u) = \dfrac{e^u}{1+e^u}.$

How do we estimate β? Consider the likelihood

$$\begin{aligned} L(y_1, \ldots, y_n) &= \prod_{i:y_i=0} (1 - F(\beta^T x_i)) \prod_{i:y_i=1} F(\beta^T x_i) \\ &= \prod_{i=1}^{n} F^{y_i}(\beta^T x_i)(1 - F(\beta^T x_i))^{1-y_i}, \end{aligned}$$

Two models (depending on F):

- Probit model $\Phi(u) = \frac{1}{\sqrt{2\pi}} \int\limits_{-\infty}^{u} e^{-\frac{z^2}{2}} dz.$
- Logit model $\Lambda(u) = \frac{e^u}{1 + e^u}.$

How do we estimate β? Consider the likelihood

$$\begin{aligned} L(y_1, \ldots, y_n) &= \prod_{i:y_i=0} (1 - F(\beta^T x_i)) \prod_{i:y_i=1} F(\beta^T x_i) \\ &= \prod_{i=1}^{n} F^{y_i}(\beta^T x_i)(1 - F(\beta^T x_i))^{1-y_i}, \\ \ln L(y_1, \ldots, y_n) &= \sum_{i=1}^{n} \left(y_i \ln F(\beta^T x_i) + (1 - y_i) \ln(1 - F(\beta^T x_i)) \right), \end{aligned}$$

Two models (depending on F):

- Probit model $\Phi(u) = \frac{1}{\sqrt{2\pi}} \int\limits_{-\infty}^{u} e^{-\frac{z^2}{2}} dz.$
- Logit model $\Lambda(u) = \frac{e^u}{1+e^u}.$

How do we estimate β? Consider the likelihood

$$\begin{aligned} L(y_1, \ldots, y_n) &= \prod_{i:y_i=0} (1 - F(\beta^T x_i)) \prod_{i:y_i=1} F(\beta^T x_i) \\ &= \prod_{i=1}^{n} F^{y_i}(\beta^T x_i)(1 - F(\beta^T x_i))^{1-y_i}, \\ \ln L(y_1, \ldots, y_n) &= \sum_{i=1}^{n} \left(y_i \ln F(\beta^T x_i) + (1 - y_i) \ln(1 - F(\beta^T x_i)) \right), \\ \frac{\partial \ln L}{\partial \beta} &= \sum_{i=1}^{n} \left(\frac{y_i f(\beta^T x_i)}{F(\beta^T x_i)} - \frac{(1 - y_i) f(\beta^T x_i)}{1 - F(\beta^T x_i)} \right) x_i = 0, \end{aligned}$$

where f is a pdf.

For the logit model:

$$\frac{\partial \ln L}{\partial \beta} = \sum_{i=1}^{n} (y_i - \Lambda(\beta^T x_i)) x_i = 0.$$

For the logit model:

$$\frac{\partial \ln L}{\partial \beta} = \sum_{i=1}^{n} (y_i - \Lambda(\beta^T x_i)) x_i = 0.$$

For the probit model:

$$\frac{\partial \ln L}{\partial \beta} = \sum_{i=1}^{n} \frac{(2y_i - 1)\varphi(\beta^T x_i)}{\Phi((2y_i - 1)\beta^T x_i)} x_i = 0, \quad \varphi(x) = \Phi'(x).$$

Using R...

glm: logit model

```
glm(formula, family = binomial(”logit”), data, weights, subset, na.action,
start = NULL, etastart, mustart, offset, control = list(...), model = TRUE,
method = ”glm.fit”, x = FALSE, y = TRUE, contrasts = NULL, ...)
```

glm: probit model

```
glm(formula, family = binomial(”probit”), data, weights, subset,
na.action, start = NULL, etastart, mustart, offset, control = list(...),
model = TRUE, method = ”glm.fit”, x = FALSE, y = TRUE, contrasts =
NULL, ...)
```

Using R...

glm: logit model

```
glm(formula, family = binomial(”logit”), data, weights, subset, na.action,
start = NULL, etastart, mustart, offset, control = list(...), model = TRUE,
method = ”glm.fit”, x = FALSE, y = TRUE, contrasts = NULL, ...)
```

glm: probit model

```
glm(formula, family = binomial(”probit”), data, weights, subset,
na.action, start = NULL, etastart, mustart, offset, control = list(...),
model = TRUE, method = ”glm.fit”, x = FALSE, y = TRUE, contrasts =
NULL, ...)
```

summary

```
summary(object)
```

confint

confint(object)

confint

```
confint(object)
```

wald.test

```
library(aod)
wald.test(Sigma, b, Terms = NULL, L = NULL, H0 = NULL, df = NULL,
verbose = FALSE)
vcov(object)
coef(object)
```

Consider example [http://www.ats.ucla.edu/stat/r/dae/logit.htm].
There are 4 variables.
There are 3 independent variables: mark at the exam (`gre`), mean mark (at school) (`gpa`), rank of the school (`rank`).
The dependent variable is the results of the exam: accept or reject at the University (`admit`).
Variables `gre` and `gpa` are continuous, variable `rank` is discrete (values from 1 (best school) to 4 (worst school).
Get data from website and print first 6 strings of dataset:

```
mydata <- read.csv("http://www.ats.ucla.edu/stat/data/
binary.csv")
head(mydata)
```

Result

```
    admit  gre   gpa  rank
1       0  380  3.61     3
2       1  660  3.67     3
3       1  800  4.00     1
4       1  640  3.19     4
5       0  520  2.93     4
6       1  760  3.00     2
```

For descriptive statistics we use function `summary`:

```
summary(mydata)
```

Result

```
     admit             gre             gpa             rank      
 Min.   :0.0000   Min.   :220.0   Min.   :2.260   Min.   :1.000  
 1st Qu.:0.0000   1st Qu.:520.0   1st Qu.:3.130   1st Qu.:2.000  
 Median :0.0000   Median :580.0   Median :3.395   Median :2.000  
 Mean   :0.3175   Mean   :587.7   Mean   :3.390   Mean   :2.485  
 3rd Qu.:1.0000   3rd Qu.:660.0   3rd Qu.:3.670   3rd Qu.:3.000  
 Max.   :1.0000   Max.   :800.0   Max.   :4.000   Max.   :4.000  
```

```
sapply(mydata, sd)
```

Result

```
      admit         gre         gpa        rank 
  0.4660867 115.5165364   0.3805668   0.9444602 
```

For binary regression we use function `glm`. Before this, we convert rank using function `factor` to make it categoric variable.

The name of the model is `mylogit`.

Main arguments are “formula”: `admit ~ gre + gpa + rank`, name of dataset: `data = mydata`, and type of regression model: `family = binomial("logit")`:

```
mydata$rank <- factor(mydata$rank)
mylogit <- glm(admit ~ gre + gpa + rank, data = mydata,
family = binomial("logit"))
```

summary(mylogit)

Results:

```
Call:
glm(formula = admit ~ gre + gpa + rank, family = binomial("logit"),
data = mydata)
Deviance Residuals:
    Min        1Q      Median      3Q       Max
 -1.6268   -0.8662   -0.6388   1.1490    2.0790
Coefficients:
               Estimate   Std. Error   z value   Pr(>|z|)
(Intercept)   -3.989979    1.139951    -3.500    0.000465   ***
    gre        0.002264    0.001094     2.070    0.038465    *
    gpa        0.804038    0.331819     2.423    0.015388    *
 rank[T.2]    -0.675443    0.316490    -2.134    0.032829    *
 rank[T.3]    -1.340204    0.345306    -3.881    0.000104   ***
 rank[T.4]    -1.551464    0.417832    -3.713    0.000205   ***
```

```
— Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
(Dispersion parameter for binomial family taken to be 1)
Null deviance: 499.98 on 399 degrees of freedom
Residual deviance: 458.52 on 394 degrees of freedom
AIC: 470.52
Number of Fisher Scoring iterations: 4.
```

Estimates of coefficients of logistic regression are presented in Table, where one can find the values of Wald statistics and corresponding p-values.

By default, $\alpha = 0.05$.

All coefficients are significant for the example.

We may use function `confint` for construction of confidence intervals with confidence level 0.95.

```
confint(mylogit)
```

Results

```
                  2.5 %    97.5 %
 (Intercept)  -6.271620  -1.79255
gre            0.000138   0.00444
gpa            0.160296   1.46414
rank2         -1.300889  -0.05675
rank3         -2.027671  -0.67037
rank4         -2.400027  -0.75354
```

Maximal likelihood method is used for confidence intervals.

Likelihood ratio statistics is equal `Null deviance` − `Residual deviance` and it is $499.98 - 458.52 = 41.46$.

If the null hypotesis (about non-significance of logistic regression) is true, statistics of the test is distributed by χ^2-distribution with $399 - 394 = 5$ degrees of freedom.

Critical range is $(11.0705, \infty)$.

Value `AIC` can be used for model fitness. The less the value, the better the model.

One may use Wald test for estimation of logit model with function `wald.test` from package `aod`:

```
install.packages(”aod”)
library(aod)
wald.test(b = coef(mylogit), Sigma = vcov(mylogit), Terms = 1:6)
```

By `Terms` the list of independent variables to be tested is given.

`wald.test`:

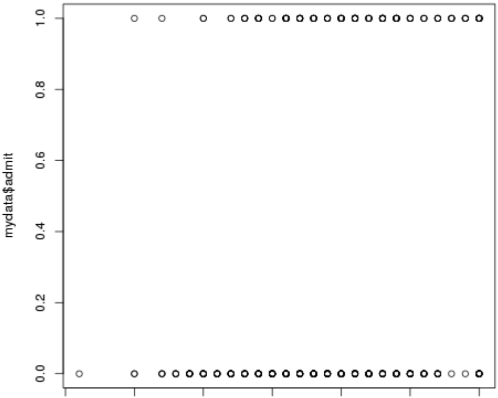
Wald test:

Chi-squared test:
$X2 = 73.4,\ df = 6,\ P(> X2) = 8.2e - 14$

Wald statistics is 73.4. The p-value is `8.2e-14`, less than 0.05. We use coefficients of logistic regression as `b`, and `vcov(mylogit)` as covariance matrix. As `Terms` we use independent variables 1–6.

For graph of admit as a function of gre, and admit as a function of gpa we use plot:

```
plot(mydata$admit,mydata$gre)
plot(mydata$admit,mydata$gpa)
```



Probit model

myprobit < − glm(admit ∼ gre + gpa + rank, data = mydata, family = binomial(”probit”))
summary(myprobit)

Results:

glm(formula = admit ∼ gre + gpa + rank, family = binomial(”probit”), data = mydata)

Deviance Residuals:

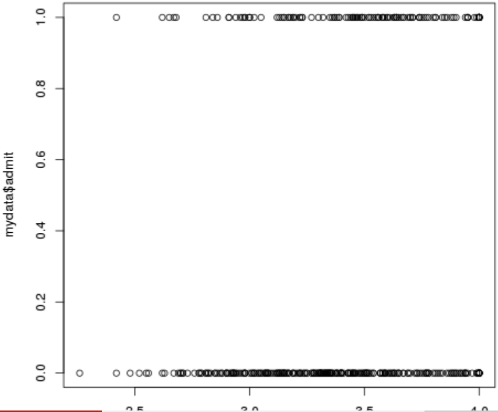
```
    Min        1Q      Median      3Q       Max
 -1.6163   -0.8710   -0.6389    1.1560   2.1035
```

```
Coefficients:
              Estimate    Std. Error   z value   Pr(>|z|)
 (Intercept)  -2.386836   0.673946     -3.542    0.000398   ***
     gre       0.001376   0.000650      2.116    0.034329    *
     gpa       0.477730   0.197197      2.423    0.015410    *
  rank[T.2]   -0.415399   0.194977     -2.131    0.033130    *
  rank[T.3]   -0.812138   0.208358     -3.898    9.71e-05   ***
  rank[T.4]   -0.935899   0.245272     -3.816    0.000136   ***
------------
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
(Dispersion parameter for binomial family taken to be 1)
Null deviance: 499.98 on 399 degrees of freedom
Residual deviance: 458.41 on 394 degrees of freedom
AIC: 470.41
Number of Fisher Scoring iterations: 4
```

The model is significant.

For probit model the maximal likelihood statistics is 41.57.

If the null hypotesis (about non-significance of probit regression) is true, statistics of the test is distributed by χ^2-distribution with 5 degrees of freedom. Critical range is $(11.0705, \infty)$.



VIF (test for multicollinearity)

Like in case of linear regression, we should check for multicollinearity in the model.

```
vif(mylogit)
vif(myprobit)
```

Value `AIC` for probit model is less than for logit model, therefore, probit model is better according to this criteria.

Confidence intervals for probit model:

```
confint(myprobit)
```

Result:

```
                   2.5 %       97.5 %
(Intercept)  -3.7201050682  -1.076327713
gre           0.0001104101   0.002655157
gpa           0.0960654793   0.862610221
rank[T.2]    -0.7992113929  -0.032995019
rank[T.3]    -1.2230955861  -0.405008112
rank[T.4]    -1.4234218227  -0.459538829
```

Verify the probit model using function `wald.test`:

```
library(aod)
wald.test(b = coef(myprobit), Sigma = vcov(myprobit), Terms = 1:6)
```

Result:

Wald test:

———-

Chi-squared test:
X2 = 83.6, df = 6, $P(> X2) = 6.7e - 16$

The value of Wald statistics is 83.6. The p-value is $6.7e - 16$ which is less 0.05.

Predictions

predicted $< -$ predict(mylogit, testData, type=”response”)

When we use the `predict` function on this model, it will predict the log(odds) of the Y variable. This is not what we ultimately want because, the predicted values may not lie within the 0 and 1 range as expected. So, to convert it into prediction probability scores that is bound between 0 and 1, we use the `plogis()`.

Predictions

predicted $< -$ plogis(predict(mylogit, testData))

Decide on optimal prediction probability cutoff for the model

- The default cutoff prediction probability score is 0.5 or the ratio of 1's and 0's in the training data. But sometimes, tuning the probability cutoff can improve the accuracy in both the development and validation samples.
- The `InformationValue::optimalCutoff` function provides ways to find the optimal cutoff to improve the prediction of 1's, 0's, both 1's and 0's and to reduce the misclassification error.
- Compute the optimal score that minimizes the misclassification error for the above model.

CutOff

```
library(InformationValue)
optCutOff < − optimalCutoff(testData$admit, predicted)[1]
```

Confusion Matrix

Predictions vs. Observed	1	0
1	TP	FP
0	FN	TN

- TP: True Positive,
- FP: False Positive,
- FN: False Negative,
- TN: True Negative.

confusion_matrix(mylogit,DATA=NA)

where `DATA` is a data frame on which the confusion matrix will be made. If omitted, the confusion matrix is on the data used in $mylogit$.

Confusion Matrix

If specified, the data frame must have the same column names as the data used to build the model in $mylogit$. This function makes classifications on the data used to build a logistic regression model by predicting dependent variable when the predicted probability exceeds 50%.

```
confusionMatrix(testData$admit, predicted, threshold = optCutOff)
```

Specificity and Sensitivity

Sensitivity (or True Positive Rate) is the percentage of 1's (actuals) correctly predicted by the model:

$$Sensitivity = \frac{TP}{TP + FN}$$

Specificity is the percentage of 0's (actuals) correctly predicted. Specificity can also be calculated as $1 - FalsePositiveRate$.

$$Specificity = \frac{TN}{TN + FP}$$

Sensitivity and specificity

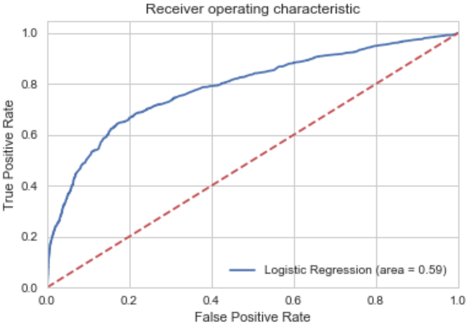
```
sensitivity(testData$admit, predicted, threshold = optCutOff)
specificity(testData$admit, predicted, threshold = optCutOff)
```

The receiver operating characteristic (ROC) curve is another common tool used with binary classifiers. Receiver Operating Characteristic(ROC) summarizes the model's performance by evaluating the trade offs between true positive rate (sensitivity) and false positive rate (1- specificity). For plotting ROC, it is advisable to assume $p > 0.5$ since we are more concerned about success rate. ROC summarizes the predictive power for all possible values of $p > 0.5$. The area under curve (AUC), referred to as index of accuracy(A) or concordance index, is a perfect performance metric for ROC curve. Higher the area under curve, better the prediction power of the model. Below is a sample ROC curve. The ROC of a perfect predictive model has TP equals 1 and FP equals 0. This curve will touch the top left corner of the graph.

ROC Curve

```
plotROC(testData$admit, predicted)
```

ROC Curve


Receiver operating characteristic
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.2
0.4
0.6
0.8
1.0
Logistic Regression (area = 0.59)

Dataset

binary regression.xls

Row: student.

First column: the number of hours the student slept before exam.

Second column: the number of hours the student studied before exam.

Third column: the exam is passed (1) or failed (0).

Make logit and probit models, write the equation of the model, test the model with different statistics.