Logistic Regression

R Tutorials for Applied Statistics

Note on required packages: The following code requires the package `erer`, `caret`, and `e1071` to compute marginal effects statistics for a logistic regression and logistic regression fit statistics. If you have not already done so, download, install, and load the library with the following code:

```r
# This only needs to be executed once for your machine
install.packages("erer")

# This only needs to be executed once for your machine
install.packages("caret")

# This only needs to be executed once for your machine
install.packages("e1071")

# This needs to be executed every time you load R
library("erer")

# This needs to be executed every time you load R
library("caret")

# This needs to be executed every time you load R
library("e1071")
```

---

1 Logistic Regression Structure

Binary variables can be used to estimate *proportions* or *probabilities* that an event will occur. If a binary variable is equal to 1 for when the event occurs, and 0 otherwise, estimates for the mean can be interpreted as the probability that the event occurs.

A **logistic regression** or **logit** is a regression model where the outcome is the probability that a binary variable takes on a value equal to 1.0. The probability for the outcome variable is explained by one or more independent variables, which themselves can be binary or continuous.

A logistic regression takes the form,

$$\log(Odds_i) = \beta_0 + \beta_1 x_{1,i} + \beta_2 x_{2,i} + \ldots + \beta_k x_{k,i} + \epsilon_i$$

log(Oddsi)=β0+β1x1,i+β2x2,i+…+βkxk,i+ϵi

where $Odds_i$Oddsi is called an **odds ratio** and is given by,

$$Odds_i = \frac{P(y_i = 1)}{P(y_i = 0)} = \frac{P(y_i = 1)}{1 - P(y_i = 1)}$$

Oddsi=P(yi=1)P(yi=0)=P(yi=1)1−P(yi=1)

The odds ratio has a range between 0 and $+\infty$ +∞, and increases as the probability that y_i yi takes on a value of 1.0 increases. The log odds ratio, $\log(Odds_i)$log(Oddsi), can take on any value between $-\infty$ −∞ and $+\infty$ +∞. The construction of the odds ratio and logarithmic operation transform our binary outcome variable to a variable with an infinite range.

2 Example: Mortgage loan applications

The data set, `loanapp.RData`, includes actual data from 1,777 mortgage loan applications, including whether or not a loan was approved, and a number of possible explanatory variables including demographic information of the applicants and financial variables related to the applicant’s ability to pay the loan such as the applicant’s income and employment information, value of the mortgaged property, and credit history.

The code below loads the `R` data set, which creates a data frame called `df`, and a list of descriptions for the variables called `desc`.

```
load(url("https://murraylax.org/datasets/loanapp.RData"))
```

3 Estimating a Logistic Regression

Let us estimate a logistic regression with loan approval status as the outcome variable (`approve`) and the following explanatory variables:

- `loanprc` :Loan amount relative to price of the property
- `loaninc`: Loan amount relative to total income
- `obrat`: Value of other debt obligations relative to total income
- `mortno`: Dummy variable equal to 1 if the applicant has no previous mortgage history, 0 otherwise

- `unem`: Unemployment rate in the industry where the applicant is employment

```
lmapp <- glm(approve ~ loanprc + loaninc + obrat + mortno + unem,
             data=df, family=binomial(link="logit"), x=TRUE)
```

The function `glm()` is used to estimate *general linear models*, a large class of models for which a logistic regression is one type. The
parameter, `family=binomial(link="logit")` specifies a logistic regression model.

The parameter, `x=TRUE`, tells `glm()` to include in its return value intermediate calculations used to produce the regression results. We will not directly use these intermediate results, but we need them stored for functions that we run later in this tutorial.

We can see a summary of the regression results with a call to `summary()`.

```
summary(lmapp)
```

```
##
## Call:
## glm(formula = approve ~ loanprc + loaninc + obrat + mortno +
##     unem, family = binomial(link = "logit"), data = df, x = TRUE)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7968   0.3093   0.4499   0.5618   2.0335
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  5.6181792  0.4974805  11.293  < 2e-16 ***
## loanprc     -0.0203893  0.0045792  -4.453 8.48e-06 ***
## loaninc     -0.0003379  0.0004075  -0.829 0.406911
## obrat       -0.0549032  0.0091192  -6.021 1.74e-09 ***
## mortno       0.6140803  0.1811540   3.390 0.000699 ***
## unem        -0.0660383  0.0308781  -2.139 0.032462 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1346.3  on 1776  degrees of freedom
## Residual deviance: 1242.3  on 1771  degrees of freedom
## AIC: 1254.3
##
## Number of Fisher Scoring iterations: 5
```

Some of the coefficients have expected signs, and most are statistically significant.

- The negative sign on `loanprc` indicates that as the loan amount increases relative to the selling price of the property, the likelihood the loan will be approved decreases. The small p-value (8.48×10^{-6}) indicates there is statistical significance that loan amount as a percentage of price does influence the probability of being approved for the loan, given the other variables in the model.
- The negative sign on `loaninc` indicates that as the loan increases relative to the applicant's income, the likelihood the loan is approved decreases. However, the large p-value (0.407) suggests that given the effects of the other variables in the model, we fail to find enough statistical evidence that loan amount as a ratio of income influences the probability of a loan being approved.
- The negative sign on `obrat` indicates that as other debt obligations rise relative to the applicant's income, the likelihood the loan is approved decreases. The small p-value (1.74×10^{-9}) indicates we have statistical significance that given the effects of the other variables in the model, other debt obligations as a percentage of income does influence the probability that a loan is approved.
- The positive sign on `mortno` indicates that people who have no previous mortgage history are more likely than others to be approved for a loan. With a p-value of 0.0007, the result is statistically significant. Given all the other variables in the model, there is statistical evidence that first-time mortgage applicants are more likely to be approved for a loan than others.
- Finally, the negative sign on `unem` indicates that the unemployment rate in the industry the applicant is employed is associated with a lower probability of a loan being approved. With a p-value equal to 0.0325, there is sufficient statistical evidence to conclude the unemployment rate in the industry the applicant is employed is predictive of whether the loan will be approved.

The magnitude of the coefficients have no intuitive meaning. They are the magnitude that the *log odds ratio*increases when each explanatory variable increases by one unit.

4 Marginal Effects

The *marginal effects* of a regression are estimates of the impact that one-unit increases in the explanatory variables have on the outcome variable.

In ordinary least squared regression, the coefficients are equal to the marginal effects.

In a logistic regression, the marginal effects are defined as the impact that one-unit increases in the explanatory variables have on *the probability that the outcome variable is equal to 1.0*. The right-hand-side expression in the logistic regression model is a log-odds ratio, which is a mathematical transformation of the probability that the outcome variable is equal to 1.0. Therefore, we must take a transformation of the coefficients to obtain an estimate for the marginal effects.

The calculation of the marginal effects is further complicated in that the estimate depends on the value of the explanatory variables in the model. Unlike a simple regression where the estimate of the marginal effects is always equal to the coefficient, the estimate of the marginal effect of a logistic regression changes when any of the explanatory variables change. Rather than giving a range of marginal effects for a range of each explanatory variables, it is common to evaluate the marginal effect at the mean for every explanatory variable. The function call below does this by default.

The function `maBina()` from the package `erer` can be used to compute the marginal effects of the logistic regression as follows.

```
maBina(lmapp)
```

```
## Warning in f.xb * coef(w): Recycling array of length 1 in array-vector arithmetic is depr
##   Use c() or as.vector() instead.
```

```
##             effect error t.value p.value
## (Intercept)  0.531 0.042  12.516   0.000
## loanprc     -0.002 0.000  -4.580   0.000
## loaninc      0.000 0.000  -0.829   0.407
## obrat       -0.005 0.001  -6.203   0.000
## mortno       0.054 0.014   3.724   0.000
## unem        -0.006 0.003  -2.141   0.032
```

The output above implies the following:

- `loanprc`: A 1 percentage increase in the size of the loan relative to the price of the house leads to a decrease in the probability of approval by 0.002 (i.e. 0.2 percentage points less likely)
- `loaninc`: The point estimate equal to 0.000 indicates that an increase in the size of the loan relative to income has a zero impact on the probability of loan approval.

- `obrat`: A 1 percentage point increase in other debt obligations as a percentage of income leads to a decrease in the probability of approval by 0.005 (i.e. 0.5 percentage points less likely)
- `mortno`: A first-time mortgage borrower has a 5.4% greater chance of being approved than other borrowers.
- `unem`: A 1 percentage point increase in the unemployment rate in the applicant’s industry is associated with a 0.6% lesser chance of being approved for a loan.

5 Confusion Matrix

A confusion matrix gives a short comparison of the actual values for your outcome variable, and the values predicted from your outcome variable.

Let us first compute the predicted values from our logistic regression above.

The return value from `glm()` that we called `lmapp` includes an object called `fitted.values`. These are the predicted probabilities based on our logistic regression model that a loan is approved for each observation in the sample.

We can view the first few observations in this object with a call to `head()`:

```
head(lmapp$fitted.values)
```

```
##         2         3         4         5         6         7 
## 0.8635463 0.8866126 0.8905850 0.8425313 0.8693817 0.8224483
```

Each of the values are numbers between 0 and 1 for the probability that the outcome variable should take on a value of 1.

A confusion matrix compares these predicted values with the actual values, and reports how often we correctly identified $y = 1$ and $y = 0$.

The code below creates predicted values for y based on a cut off value of 0.5 for the predicted probability.

```
predicts <- as.numeric(lmapp$fitted.values >= 0.5)
```

The code below computes the confusion matrix along with the several other model performance measures.

```r
confusionMatrix(as.factor(predicts), as.factor(df$approve))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0    8    5
##          1  216 1548
##
##                Accuracy : 0.8756
##                  95% CI : (0.8594, 0.8906)
##     No Information Rate : 0.8739
##     P-Value [Acc > NIR] : 0.4325
##
##                   Kappa : 0.0544
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.035714
##             Specificity : 0.996780
##          Pos Pred Value : 0.615385
##          Neg Pred Value : 0.877551
##              Prevalence : 0.126055
##          Detection Rate : 0.004502
##    Detection Prevalence : 0.007316
##       Balanced Accuracy : 0.516247
##
##        'Positive' Class : 0
##
```

The first parameter to `confusionMatrix()` is the vector of predicted values and the second parameter is the vector of actual outcomes.

- When the actual value was $y = 0$ (i.e. the loan was not approved), the logistic regression predicted $\hat{y} = 0$ correctly only 8 times, and predicted $\hat{y} = 1$ incorrectly 216 times
- When the actual value was $y = 1$, the logistic regression predicted $\hat{y} = 1$ correctly 1548 times, and predicted $\hat{y} = 0$ incorrectly 5 times

Overall, the model does not seem to predict well when a loan is not approved.

Still, the overall accuracy is 87.6%. This measure is deceiving, as it may suggest this is a well-performing predictive model. However, the high accuracy is due to the high level of correctly predicting loans that are approved, which is most of the sample. The model is extremely poor at predicted when a loan is not approved.

The **No Information Rate (NIR)** is similar, 87.4%. This is the accuracy rate you would expect if you had no information except for knowing which was the more popular outcome, $y = 0$ or $y = 1$, and simply used that as your predicted value regardless of any information from the explanatory variables. Loan approvals are much more common, making up 87.4% of the sample. If you just guessed approve for every observation, you would have an “accuracy” rate of 87.4%. That the actual accuracy rate, 87.6% is not much larger, suggests that this is a poorly performing model for forecasting.

The output above also includes a hypothesis test for determining whether there is statistical evidence that the model accuracy is greater than the no information rate.

The null and alternative hypotheses are the following:

$$H_0: \text{Accuracy Rate} = \text{No information Rate}$$

$$H_A: \text{Accuracy Rate} > \text{No information Rate}$$

The p-value is equal to 0.4325. We fail to reject the null hypothesis. This means we fail to find evidence that the model predicts loan approval any better than simply guessing the more popular outcome (approval) for every observation.