Logistic Regression Software

Joseph M Hilbe September 2010

Statistics.com: Logistic Regression

Logistic regression is a statistical procedure that calculates the parameter estimates or coefficients and related standard errors and confidence intervals of a non-linear logistic model. A nice feature of the logistic model is that its exponentiated coefficients are odds ratios. Another nice feature is that the model may be estimated using traditional maximum likelihood methods, or by viewing logistic regression as a member of the family of generalized linear models. Each method has its positive and negative values.

I refer the reader to *Logistic Regression Models* (2009, Hilbe) for the derivation of the estimating methods used in both maximum likelihood and generalized linear models (GLM). Here we shall focus on the software applications that are currently in existence for performing logistic regression analysis.

GLM models such as logistic regression were originally estimated using maximum likelihood (MLE). The process was originally tedious, and required the statistician to select appropriate starting values for the coefficients, and submitting a work project to a typically off-site mainframe computer. Results would be returned the following day (hopefully). If there were any errors, or the researcher wanted to try new variables, or perhaps combine variables in the model, a new work project would have to be executed. The process was slow and demanded considerable patience. To be honest though, the statistician learned to be careful when constructing a model.

In 1972 John Nelder and Robert Wedderburn developed a method of linearizing otherwise non-linear models if they were members of the exponential family of distributions. They called the method generalized linear models. GLM is a simplification of the maximum likelihood method that is allowed due of the unique properties the exponential distribution. When logistic regression is estimated from within the GLM framework, the results are identical to those produced when the model is estimated using MLE techniques.

Several leading software packages allow estimation of logistic regression using either MLE or GLM. Remember, though, GLM is a type of MLE. GLMs are estimated using a weighted linear least squares algorithm, which takes comparatively little RAM to estimate compared to MLE, so became quite popular when RAM was not easily available.

Most software uses the value of 1 to indicate Bernoulli success, and 0 as failure, or not-success. SAS, however, uses the reverse, resulting in some confusion when comparing statistical output. Moreover, most packages assign by default the 1st level of a categorical variable as the reference; SAS assigns the highest level. Missing values are low in SAS, but high in Stata. Read the respective manuals for information on how to deal with missing values, as well as on how to amend the default assignment of reference levels.

GLM-MLE software capabilities

	GLM	MLE
Stata	glm	logit; logistic
R	glm	
SAS	Genmod	Logistic
SPSS	Genlin	logistic
LIMDEP		logit
Statistica	qlz	
LogXact		logistic
Systat		logit
NCSS		logistic regression

SPSS, Limdep, Statistica, Systat, and NCSS are menu driven systems. The first three allow modeling from a command line, but it is rarely used. SAS and Stata are mostly command line oriented, although Stata also has a full-fledged menu system. R is completely command driven, except for user functions that have been created for that purpose. Most R menu functions relate to graphics, however, not modeling.

Stata's *logit* command is one of the oldest commands in Stata. It is a MLE program, written entirely in C. The basic command logic is:

```
logit y  xvars, <or>
```

The *or* option produces odds ratios rather than the default coefficients. I wrote the first *logistic* command in Stata back in 1990. The command uses Stata's higher programming language as a wrapper, calling *logit* for the estimation. The *logistic* command displays odds ratios as the default output, and provides a host of residuals as options. Stata now provides the same residuals for *logit*, but as post-estimation commands. The key difference in *logit* and *logistic* is that *logistic* command incorporates m-asymptotics; ie. the residuals and fit statistics are based on covariate patterns and not on individual observations. This is a more accurate method of understanding residuals and produces more accurate fit tests. *logistic* also has options for the Hosmer-Lemeshow goodness-of-fit test, classification analysis, ROC analysis, and other tests.

Stata's *glm* command, R's *glm* command, SAS's *Genmod* procedure, SPSS's *Genlin* procedure, and Statistica's *glz* command are all GLM applications. GLM allows the fitting of a number of

models based on the model's underlying probability distribution. For instance, a binary response logistic regression – the standard logistic regression paradigm – is based on the Bernoulli distribution, which itself is a type of binomial distribution. The binomial distribution has a mean parameter that is divisible into two components, a numerator – the number of successes for a given pattern of covariates, and a denominator – the total number of observations having the same covariate pattern. This manner of parameterizing the logistic model is also called grouped logistic regression.

The relationship of grouped models (binomial) and observation based can be exemplified using the table below

num	denom	x1	x2
1	3	1	1
3	5	0	1
0	4	1	0
5	5	0	0

The first of four observations tells us one out of three cases having values of 1 for both predictors is a success (1). There are 2 failures, which are not shown. For the fourth case all five observations are successes, and all five have 0's as values for both predictors.

In Stata we would model the above as:

```
glm num x1 x2, fam(bin denom)
```

If the above grouped data is converted to observation level, it would appear as:

y	x1	x2
1	1	1
0	1	1
0	1	1
1	0	1
1	0	1
1	0	1
0	0	1
0	0	1
0	1	0
.	.	.
0	0	0

Which can be seen as identical to the grouped format. We model this in Stata as

```
glm x1 x2, fam(bin)
```

To obtain odds ratios, *glm* uses the *eform* option. For example

```
glm x1 x2, fam(bin) eform
```

In order to obtain fitted values, we type after the *glm* command has been run

```
predict mu
```

for predicted values. For logistic regression these are probabilities ranging from 0 to 1. They give the probability of success for an observation having that specific covariate pattern. I used the word *mu* with the predict command. Any legal name is OK. The fitted values are then stored in the variable by that name.

In R, we use the following commands to obtain a table of observation-level parameter estimates:

```
myglm <- glm(y ~ x1 + x2, family=binomial(link=logit))
summary(myglm)
```

For grouped data, the R *glm* function requires that the response terms be set up as success and failure. A variable with the number of success – call it *y*, and a variable with the number of failures – call it *noty*, must be created and set up in the *glm* algorithm as:

```
noty <-  denom – num
mygglm <- glm(cbind(y,noty) ~ x1 + x2, family=binomial(link=logit))
summary(mygglm)
```

Since, like Stata, the *logit* link is the default link for the binomial family, the link function does not have to be specified. So the following is fine.

```
noty <-  denom – num
mygglm <- glm(cbind(y,noty) ~ x1 + x2, family=binomial)
summary(mygglm)
```

If we want to have odds ratios in place of coefficients, and corresponding confidence intervals, two additional lines need to be added to the R code:

```
exp(coef(mygglm))     # odds ratios
exp(confint(mygglm))  # Confidence intervals of odds ratios
```