

## Using Stata 11 & 12 for Logistic Regression

**NOTE:** The routines `spost9`, `lrdrop1`, and `extremes` are used in this handout. Use the `findit` command to locate and install them. See related handouts for the statistical theory underlying logistic regression. Most but not all of the commands shown in this handout will also work in earlier versions of Stata, but the syntax is sometimes a little different.

**Commands.** Stata and SPSS differ a bit in their approach, but both are quite competent at handling logistic regression. With large data sets, I find that Stata tends to be far faster than SPSS, which is one of the many reasons I prefer it.

Stata has various commands for doing logistic regression. They differ in their default output and in some of the options they provide. My personal favorite is `logit`.

```
. use "http://www.nd.edu/~rwilliam/stats2/statafiles/logist.dta", clear
. logit grade gpa tuce psi
```

```
Iteration 0: log likelihood = -20.59173
[intermediate iterations deleted]
Iteration 5: log likelihood = -12.889633
```

```
Logit estimates                    Number of obs   =          32
                                   LR chi2(3)        =          15.40
                                   Prob > chi2         =          0.0015
                                   Pseudo R2           =          0.3740

Log likelihood = -12.889633
```

```
-----+-----
      grade |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-----+-----
      gpa   |   2.826113   1.262941     2.24   0.025   .3507938   5.301432
      tuce  |   .0951577   .1415542     0.67   0.501  - .1822835   .3725988
      psi   |   2.378688   1.064564     2.23   0.025   .29218     4.465195
      _cons |  -13.02135   4.931325    -2.64   0.008  -22.68657  -3.35613
-----+-----
```

Note that the log likelihood for iteration 0 is  $LL_0$ , i.e. it is the log likelihood when there are no explanatory variables in the model - only the constant term is included. The last log likelihood reported is  $LL_M$ . From these we easily compute the key Deviance measures:

$$DEV_0 = -2LL_0 = -2 * -20.59173 = 41.18$$

$$DEV_M = -2LL_M = -2 * -12.889633 = 25.78$$

Also note that the default output does not include  $\exp(b)$ . To get that, include the `or` parameter (`or = odds ratios =  $\exp(b)$` ). (Or, you can use the `logistic` command, which reports  $\exp(b)$  (odds ratios) by default.)

```
. logit grade gpa tuce psi, or nolog
```

```
Logit estimates                                     Number of obs   =          32
                                                    LR chi2(3)      =          15.40
                                                    Prob > chi2     =          0.0015
Log likelihood = -12.889633                       Pseudo R2      =          0.3740
```

grade	Odds Ratio	Std. Err.	z	P> z	[95% Conf. Interval]
gpa	16.87972	21.31809	2.24	0.025	1.420194 200.6239
tuce	1.099832	.1556859	0.67	0.501	.8333651 1.451502
psi	10.79073	11.48743	2.23	0.025	1.339344 86.93802

There are various other options of possible interest, e.g. just as with OLS regression you can specify robust standard errors, change the confidence interval and do stepwise logistic regression.

You can further enhance the functionality of Stata by downloading and installing `spost9` (which includes several post-estimation commands) and `lrdrop1`. Use the `findit` command to get these. The rest of this handout assumes these routines are installed, so if a command isn't working, it is probably because you have not installed it.

**Hypothesis testing.** Stata makes you go to a little more work than SPSS does to make contrasts between nested models. You need to use the `estimates store` and `lrtest` commands. Basically, you estimate your models, store the results under some arbitrarily chosen name, and then use the `lrtest` command to contrast models. For example,

```
. * Model 0: Intercept only
. quietly logit grade

. est store M0

. * Model 1: GPA added
. quietly logit grade gpa

. est store M1

. * Model 2: GPA + TUCE
. quietly logit grade gpa tuce

. est store M2

. * Model 3: GPA + TUCE + PSI
. quietly logit grade gpa tuce psi

. est store M3

. * Model 1 versus Model 0
. lrtest M1 M0

likelihood-ratio test                               LR chi2(1) =          8.77
(Assumption: M0 nested in M1)                     Prob > chi2 =          0.0031

. * Model 2 versus Model 1
. lrtest M2 M1

likelihood-ratio test                               LR chi2(1) =          0.43
(Assumption: M1 nested in M2)                     Prob > chi2 =          0.5096
```

```

. * Model 3 versus Model 2
. lrtest M3 M2

likelihood-ratio test          LR chi2(1) =      6.20
(Assumption: M2 nested in M3) Prob > chi2 =    0.0127

. * Model 3 versus Model 0
. lrtest M3 M0

likelihood-ratio test          LR chi2(3) =     15.40
(Assumption: M0 nested in M3) Prob > chi2 =    0.0015

```

Also note that the output includes z values for each coefficient (where  $z = \text{coefficient} / \text{standard error}$ ). SPSS reports these values squared and calls them Wald statistics. Technically, Wald statistics are not considered 100% optimal; it is better to do likelihood ratio tests, where you estimate the constrained model without the parameter and contrast it with the unconstrained model that includes the parameter. The `lrdrop1` command makes this easy (also see the similar `bicdrop1` command if you want BIC tests instead):

```

. logit grade gpa tuce psi, nolog

Logit estimates                Number of obs =      32
                               LR chi2(3)      =     15.40
                               Prob > chi2     =     0.0015
Log likelihood = -12.889633     Pseudo R2    =     0.3740

```

grade	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]
gpa	2.826113	1.262941	2.24	0.025	.3507938 5.301432
tuce	.0951577	.1415542	0.67	0.501	-.1822835 .3725988
psi	2.378688	1.064564	2.23	0.025	.29218 4.465195
_cons	-13.02135	4.931325	-2.64	0.008	-22.68657 -3.35613

```

. lrdrop1
Likelihood Ratio Tests: drop 1 term
logit regression
number of obs = 32

-----+-----
      grade      Df      Chi2      P>Chi2      -2*log ll      Res. Df      AIC
-----+-----
Original Model                                25.78          28          33.78
  -gpa           1           6.78          0.0092          32.56          27          38.56
  -tuce          1           0.47          0.4912          26.25          27          32.25
  -psi           1           6.20          0.0127          31.98          27          37.98
-----+-----

```

Terms dropped one at a time in turn.

You can also use the `test` command for hypothesis testing, but the Wald tests that are estimated by the `test` command are considered inferior to estimating separate models and then doing LR chi-square contrasts of the results (although in practice it often makes little difference).

```

. test psi

( 1)  psi = 0

      chi2( 1) =      4.99
Prob > chi2 =    0.0255

```

Also, Stata 9 added the `nestreg` prefix. This makes it easy to estimate a sequence of nested models and do chi-square contrasts between them. The `lr` option tells `nestreg` to do likelihood ratio tests rather than Wald tests. This can be more time-consuming but is also more accurate. The `store` option is optional but, in this case, will store the results of each model as `m1`, `m2`, etc. This would be handy if, say, you wanted to do a chi-square contrast between model 3 and model 1.

```
. nestreg, lr store(m): logit grade gpa tuce psi
[intermediate output deleted]
```

Block	LL	LR	df	Pr > LR	AIC	BIC
1	-16.2089	8.77	1	0.0031	36.4178	39.34928
2	-15.99148	0.43	1	0.5096	37.98296	42.38017
3	-12.88963	6.20	1	0.0127	33.77927	39.64221

```
. lrtest m3 m1
```

```
Likelihood-ratio test                               LR chi2(2) =      6.64
(Assumption: m1 nested in m3)                       Prob > chi2 =    0.0362
```

Also, you don't have to enter variables one at a time; by putting parentheses around sets of variables, they will all get entered in the same block.

```
. nestreg, lr: logit grade gpa (tuce psi)
[intermediate output deleted]
```

Block	LL	LR	df	Pr > LR	AIC	BIC
1	-16.2089	8.77	1	0.0031	36.4178	39.34928
2	-12.88963	6.64	2	0.0362	33.77927	39.64221

Note that AIC and BIC are reported. These are also useful statistics for comparing models, but I won't talk about them in this handout. Adding the `stats` option to `lrtest` will also cause these statistics to be reported, e.g.

```
. lrtest m3 m1, stats
```

```
Likelihood-ratio test                               LR chi2(2) =      6.64
(Assumption: m1 nested in m3)                       Prob > chi2 =    0.0362
```

Model	Obs	ll (null)	ll (model)	df	AIC	BIC
m1	32	-20.59173	-16.2089	2	36.4178	39.34928
m3	32	-20.59173	-12.88963	4	33.77927	39.64221

$R^2$  analogs and goodness of fit measures. Although it is not clearly labeled, the Pseudo  $R^2$  reported by Stata is McFadden's  $R^2$ , which seems to be the most popular of the many alternative measures that are out there. One straightforward formula is

$$Pseudo R^2 = 1 - \frac{LL_M}{LL_0} = 1 - \frac{-12.889633}{-20.59173} = 1 - .625961636 = .374$$

You can also get a bunch of other pseudo  $R^2$  measures and goodness of fit statistics by typing `fitstat` (part of the `spost9` routines) after you have estimated a logistic regression:

```
. fitstat
```

```
Measures of Fit for logit of grade
```

Log-Lik Intercept Only:	-20.592	Log-Lik Full Model:	-12.890
D(28):	25.779	LR(3):	15.404
		Prob > LR:	0.002
McFadden's R2:	0.374	McFadden's Adj R2:	0.180
Maximum Likelihood R2:	0.382	Cragg & Uhler's R2:	0.528
McKelvey and Zavoina's R2:	0.544	Efron's R2:	0.426
Variance of y*:	7.210	Variance of error:	3.290
Count R2:	0.813	Adj Count R2:	0.455
AIC:	1.056	AIC*n:	33.779
BIC:	-71.261	BIC':	-5.007

To get the equivalent of SPSS's classification table, you can use the `estat clas` command (`lstat` also works). This command shows you how many cases were classified correctly and incorrectly, using a cutoff point of 50% for the predicted probability.

```
. estat class
```

```
Logistic model for grade
```

Classified	----- True -----		Total
	D	~D	
+	8	3	11
-	3	18	21
Total	11	21	32

```
Classified + if predicted Pr(D) >= .5
True D defined as grade != 0
```

Sensitivity	Pr( +   D)	72.73%
Specificity	Pr( -   ~D)	85.71%
Positive predictive value	Pr( D   +)	72.73%
Negative predictive value	Pr( ~D   -)	85.71%
False + rate for true ~D	Pr( +   ~D)	14.29%
False - rate for true D	Pr( -   D)	27.27%
False + rate for classified +	Pr( ~D   +)	27.27%
False - rate for classified -	Pr( D   -)	14.29%
Correctly classified		81.25%

**Predicted values.** Stata makes it easy to come up with the predicted values for each case. You run the logistic regression, and then use the `predict` command to compute various quantities of interest to you.

```

. quietly logit grade gpa tuce psi

. * get the predicted log odds for each case
. predict logodds, xb

. * get the odds for each case
. gen odds = exp(logodds)

. * get the predicted probability of success
. predict p, p

. list grade gpa tuce psi logodds odds p

```

```

+-----+
| grade  gpa  tuce  psi  logodds  odds  p |
+-----+
1. |    0   2.06   22    1  -2.727399  .0653891  .0613758 |
2. |    1   2.39   19    1  -2.080255  .1248984  .1110308 |
3. |    0   2.63   20    0  -3.685518  .0250842  .0244704 |
[Intermediate results deleted]
+-----+
31. |    1     4   23    1   2.850418  17.295  .9453403 |
32. |    1   3.92   29    0   .8165872  2.262764  .6935114 |
+-----+

```

**Hypothetical values.** Stata also makes it very easy to plug in hypothetical values. One way to do this is with the `adjust` command. Here we use Stata to compute the log odds, odds, and probability of success for a hypothetical student with a gpa of 3.0 and a tuce score of 20 who is either in psi or not in psi.

```

* Log odds
. quietly logit grade gpa tuce psi
. adjust gpa=3 tuce=20, by(psi)

```

```

-----
Dependent variable: grade      Command: logit
Covariates set to value: gpa = 3, tuce = 20
-----

```

```

-----
psi |      xb
-----+-----
0 | -2.63986
1 | -.261168
-----

```

Key: xb = Linear Prediction

```

* Odds
. adjust gpa=3 tuce=20, by(psi) exp

```

```

-----
Dependent variable: grade      Command: logit
Covariates set to value: gpa = 3, tuce = 20
-----

```

```

-----
psi | exp(xb)
-----+-----
0 | .071372
1 | .770151
-----

```

Key: exp(xb) = exp(xb)

```
* Probability of getting an A
. adjust gpa=3 tuce=20, by(psi) pr
```

```
-----
Dependent variable: grade      Command: logit
Covariates set to value: gpa = 3, tuce = 20
-----
```

```
-----
psi |          pr
-----+-----
  0 |    .066617
  1 |    .435077
-----
```

```
Key:  pr = Probability
```

This hypothetical, about average student would have less than a 7% chance of getting an A in the traditional classroom, but would have almost a 44% chance of an A in a psi classroom.

Now, consider a strong student with a 4.0 gpa and a tuce of 25:

```
* Log odds
. adjust gpa=4 tuce=25, by(psi)
```

```
-----
Dependent variable: grade      Command: logit
Covariates set to value: gpa = 4, tuce = 25
-----
```

```
-----
psi |          xb
-----+-----
  0 |    .662045
  1 |    3.04073
-----
```

```
Key:  xb = Linear Prediction
```

```
* Odds
. adjust gpa=4 tuce=25, by(psi) exp
```

```
-----
Dependent variable: grade      Command: logit
Covariates set to value: gpa = 4, tuce = 25
-----
```

```
-----
psi |    exp(xb)
-----+-----
  0 |    1.93875
  1 |   20.9206
-----
```

```
Key:  exp(xb) = exp(xb)
```

```
* Probability of getting an A
. adjust gpa=4 tuce=25, by(psi) pr
```

```
-----
Dependent variable: grade      Command: logit
Covariates set to value: gpa = 4, tuce = 25
-----
```

```
-----
      psi |          pr
-----+-----
      0 |      .65972
      1 |      .954381
-----
```

```
Key: pr = Probability
```

This student has about a 2/3 chance of an A in a traditional classroom, and a better than 95% chance of an A in psi.

The `predict` command can also be used to plug in hypothetical values. In general, the `predict` command calculates values for ALL observations currently stored in memory, whether they were used in fitting the model or not. Hence, one of many possible strategies is to run the logistic regression; `preserve` the real data and then temporarily delete it; interactively enter the hypothetical data; use the `predict` and/or `gen` commands to compute the new variables; `list` the results; and finally, `restore` the original data.

```
. quietly logit grade gpa tuce psi

. * Preserve the data so we can restore it later
. preserve

* Temporarily drop all cases - the last character in the next command is
* a lowercase L, which means last case.
. drop in 1/l
(32 observations deleted)

. edit
- preserve
* I interactively entered the values you see below.

. list

      +-----+
      | grade  gpa  tuce  psi |
      +-----+
1.   |      .    3    20    0 |
2.   |      .    3    20    1 |
3.   |      .    4    25    0 |
4.   |      .    4    25    1 |
      +-----+

. predict logodds, xb
. gen odds = exp(logodds)
. predict p, p
```

```
. list
```

```
-----+-----  
| grade  gpa  tuce  psi   logodds   odds       p |  
-----+-----  
1. |      .    3    20    0   -2.639856   .0713715   .066617 |  
2. |      .    3    20    1   -0.2611683  .7701513   .4350765 |  
3. |      .    4    25    0    .6620453   1.938754   .6597197 |  
4. |      .    4    25    1    3.040733   20.92057   .9543808 |  
-----+-----
```

```
. * Restore original data  
. restore
```

Long & Freese's `spost` commands provide several other good ways of performing these sorts of tasks; see, for example, the `prvalue` and `prtab` commands. Stata 11 and higher have the `margins` command; while a little more difficult to use, it is more powerful and can easily deal with complicated situations where `adjust` and other commands might get things wrong. I show examples later in this handout.

**Stepwise Logistic Regression.** This works pretty much the same way it does with OLS regression. However, by adding the `lr` parameter, we force Stata to use the more accurate (and more time-consuming) Likelihood Ratio tests rather than Wald tests when deciding which variables to include. (Note: stepwise is available in earlier versions of Stata but the syntax is a little different.)

```
. sw, pe(.05) lr: logit grade gpa tuce psi
```

```
LR test                begin with empty model  
p = 0.0031 < 0.0500   adding  gpa  
p = 0.0130 < 0.0500   adding  psi
```

```
Logistic regression                Number of obs   =          32  
LR chi2(2)                         =          14.93  
Prob > chi2                         =          0.0006  
Log likelihood = -13.126573          Pseudo R2      =          0.3625
```

```
-----+-----  
| grade |      Coef.  Std. Err.   z    P>|z|    [95% Conf. Interval]  
-----+-----  
| gpa   |  3.063368   1.22285    2.51  0.012    .6666251    5.46011  
| psi   |  2.337776   1.040784   2.25  0.025    .2978755    4.377676  
| _cons | -11.60157   4.212904  -2.75  0.006   -19.85871   -3.344425  
-----+-----
```

**Diagnostics.** The `predict` command lets you compute various diagnostic measures, just like it does with OLS. For example, the `predict` command can generate a standardized residual. It can also generate a deviance residual (the deviance residuals identify those cases that contribute the most to the overall deviance of the model.) [WARNING: SPSS and Stata sometimes use different formulas and procedures for computing residuals, so results are not always identical across programs.]

```
. * Generate predicted probability of success  
. predict p, p
```

```
. * Generate standardized residuals  
. predict rstandard, rstandard
```

```

. * Generate the deviance residual
. predict dev, deviance

. * Use the extremes command to identify large residuals
. extremes rstandard dev p grade gpa tuce psi

```

obs:	rstandard	dev	p	grade	gpa	tuce	psi
27.	-2.541286	-1.955074	.8520909	0	3.51	26	1
18.	-1.270176	-1.335131	.5898724	0	3.12	23	1
16.	-1.128117	-1.227311	.5291171	0	3.1	21	1
28.	-.817158	-.9463985	.3609899	0	3.53	26	0
24.	-.7397601	-.8819993	.3222395	0	3.57	23	0
19.	.8948758	.9523319	.6354207	1	3.39	17	1
30.	1.060433	1.060478	.569893	1	4	21	0
15.	1.222325	1.209638	.481133	1	2.83	27	1
23.	2.154218	1.813269	.1932112	1	3.26	25	0
2.	3.033444	2.096639	.1110308	1	2.39	19	1

The above results suggest that cases 2 and 27 may be problematic. Several other diagnostic measures can also be computed.

**Multicollinearity.** Multicollinearity is a problem of the X variables, and you can often diagnose it the same ways you would for OLS. Phil Ender's `collin` command is very useful for this:

```

. collin gpa tuce psi if !missing(grade)

```

**Robust standard errors.** If you fear that the error terms may not be independent and identically distributed, e.g. heteroscedasticity may be a problem, you can add the `vce(robust)` option (or the older `robust` parameter).

```

. logit grade gpa tuce psi, vce(robust)

```

```

Iteration 0: log pseudo-likelihood = -20.59173
Iteration 1: log pseudo-likelihood = -13.496795
Iteration 2: log pseudo-likelihood = -12.929188
Iteration 3: log pseudo-likelihood = -12.889941
Iteration 4: log pseudo-likelihood = -12.889633
Iteration 5: log pseudo-likelihood = -12.889633

```

```

Logit estimates                               Number of obs =          32
                                              Wald chi2(3) =          9.36
                                              Prob > chi2 =         0.0249
Log pseudo-likelihood = -12.889633          Pseudo R2 =         0.3740

```

grade	Coef.	Robust Std. Err.	z	P> z	[95% Conf. Interval]	
gpa	2.826113	1.287828	2.19	0.028	.3020164	5.35021
tuce	.0951577	.1198091	0.79	0.427	-.1396639	.3299793
psi	2.378688	.9798509	2.43	0.015	.4582152	4.29916
_cons	-13.02135	5.280752	-2.47	0.014	-23.37143	-2.671264

Note that, in this case, the standard errors have changed very little, but sometimes the change will be much greater. Also, Stata now reports “pseudo-likelihoods” and a Wald chi-square instead of a likelihood ratio chi-square for the model. For now I won’t try to explain why. Stata will surprise you some times with the statistics it reports, but it generally seems to have a good reason for them (although you may have to spend a lot of time reading through the manuals or the online FAQs to figure out what it is.)

**Factor Variables.** Factor variables were introduced in Stata 11. Slightly modifying our earlier example,

```
. logit grade gpa tuce i.psi, nolog
```

Logistic regression

Number of obs	=	32
LR chi2(3)	=	15.40
Prob > chi2	=	0.0015
Pseudo R2	=	0.3740

Log likelihood = -12.889633

grade	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]
gpa	2.826113	1.262941	2.24	0.025	.3507938 5.301432
tuce	.0951577	.1415542	0.67	0.501	-.1822835 .3725988
1.psi	2.378688	1.064564	2.23	0.025	.29218 4.465195
_cons	-13.02135	4.931325	-2.64	0.008	-22.68657 -3.35613

The `i.psi` notation tells Stata that `psi` is a categorical variable rather than continuous. As the Stata 11 User Manual explains (section 11.4.3.1), “`i.group` is called a factor variable, although more correctly, we should say that `group` is a categorical variable to which factor-variable operators have been applied...When you type `i.group`, it forms the indicators for the unique values of `group`.”

In other words, Stata, in effect, creates dummy variables coded 0/1 from the categorical variable. In this case, of course, `psi` is already coded 0/1 – but `margins` and other post-estimation commands still like you to use the `i.` notation so they know the variable is categorical (rather than, say, being a continuous variable that just happens to only have the values of 0/1 in this sample). But if, say, we had the variable `psi` coded 1 = traditional, 2 = `psi`, the new variable would be coded 0 = traditional, 1 = `psi`.

Or, if the variable `religion` was coded 1 = Catholic, 2 = Protestant, 3 = Jewish, 4 = Other, saying `i.religion` would cause Stata to create three 0/1 dummies. By default, the first category (in this case Catholic) is the reference category, but we can easily change that, e.g. `ib2.religion` would make Protestant the reference category, or `ib(last).religion` would make the last category (Other) the reference.

At first glance, this might seem like it is only a minor convenience. Stata is saving you the trouble of having to compute dummy variables yourself. However, the advantages become much greater once you start including interaction terms or squared terms. The advantages are greater still if you start using `margins` to estimate adjusted predictions and marginal effects, as well as the `marginsplot` command (introduced in Stata 12) to graph those results. (Unfortunately, not

all commands support factor variables, e.g. you can't use factor variables with the `nestreg` prefix or with the `lrdrop1` command.) For now, I will just show a simple example showing how margins can be used after the above `logit` command to reproduce the probabilities calculated earlier:

```
. margins psi, at(gpa = 3 tuce = 20)
```

```
Adjusted predictions      Number of obs   =          32
Model VCE      : OIM

Expression   : Pr(grade), predict()
at           : gpa           =           3
              tuce          =          20
```

	Margin	Delta-method Std. Err.	z	P> z	[95% Conf. Interval]	
psi						
0	.066617	.0611322	1.09	0.276	-.0531999	.1864339
1	.4350765	.1812458	2.40	0.016	.0798413	.7903118

```
. margins psi, at(gpa = 4 tuce = 25)
```

```
Adjusted predictions      Number of obs   =          32
Model VCE      : OIM

Expression   : Pr(grade), predict()
at           : gpa           =           4
              tuce          =          25
```

	Margin	Delta-method Std. Err.	z	P> z	[95% Conf. Interval]	
psi						
0	.6597197	.2329773	2.83	0.005	.2030926	1.116347
1	.9543808	.0560709	17.02	0.000	.8444837	1.064278

Additional Information. Long and Freese's `spost9` routines include several other commands that help make the results from logistic regression more interpretable. Their book is very good:

[Regression Models for Categorical Dependent Variables Using Stata, Second Edition](#), by J. Scott Long and Jeremy Freese. 2006.

The notes for my Soc 73994 class, [Categorical Data Analysis](#), contain a lot of additional information on using Stata for logistic regression and other categorical data techniques. See

<http://www.nd.edu/~rwilliam/xsoc73994/index.html>