probit — Probit regression

Syntax

probit depvar [ indepvars ] [ if ] [ in ] [ weight ] [ , options ]

options Description

Model

noconstant suppress constant term
offset(varname) include varname in model with coefficient constrained to 1
asis retain perfect predictor variables
constraints(constraints) apply specified linear constraints
collinear keep collinear variables

SE/Robust

vce(vcetype) vcetype may be oim, robust, cluster clustvar, bootstrap, or jackknife

Reporting

level(#) set confidence level; default is level(95)
nocnsreport do not display constraints
display_options control column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling

Maximization

maximize_options control the maximization process; seldom used

nocoef do not display the coefficient table; seldom used

coeflegend display legend instead of statistics

indepvars may contain factor variables; see [U] 11.4.3 Factor variables.
depvar and indepvars may contain time-series operators; see [U] 11.4.4 Time-series varlists.
bootstrap, by, fp, jackknife, mfp, mi estimate, nestreg, rolling, statsby, stepwise, and svy are allowed; see [U] 11.1.10 Prefix commands.
vce(bootstrap) and vce(jackknife) are not allowed with the mi estimate prefix; see [MI] mi estimate.
Weights are not allowed with the bootstrap prefix; see [R] bootstrap.
vce(), nocoef, and weights are not allowed with the svy prefix; see [SVY] svy.
fweights, iweights, and pweights are allowed; see [U] 11.6 weight.
nocoeff and coeflegend do not appear in the dialog box.
See [U] 20 Estimation and postestimation commands for more capabilities of estimation commands.
probit — Probit regression

Menu

Statistics > Binary outcomes > Probit regression

Description

probit fits a maximum-likelihood probit model.

If estimating on grouped data, see the bprobit command described in [R] glogit.

Several auxiliary commands may be run after probit, logit, or logistic; see [R] logistic postestimation for a description of these commands.

See [R] logistic for a list of related estimation commands.

Options

Model

noconstant, offset(varname), constraints(constraints), collinear; see [R] estimation options.

asis specifies that all specified variables and observations be retained in the maximization process.

This option is typically not specified and may introduce numerical instability. Normally probit drops variables that perfectly predict success or failure in the dependent variable along with their associated observations. In those cases, the effective coefficient on the dropped variables is infinity (negative infinity) for variables that completely determine a success (failure). Dropping the variable and perfectly predicted observations has no effect on the likelihood or estimates of the remaining coefficients and increases the numerical stability of the optimization process. Specifying this option forces retention of perfect predictor variables and their associated observations.

SE/Robust

vce(vcetype) specifies the type of standard error reported, which includes types that are derived from asymptotic theory (oim), that are robust to some kinds of misspecification (robust), that allow for intragroup correlation (cluster clustvar), and that use bootstrap or jackknife methods (bootstrap, jackknife); see [R] vce_option.

Reporting

level(#); see [R] estimation options.

cnsreport; see [R] estimation options.

display_options: nomomitted, vsquish, noemptycells, baselevels, allbaselevels, nofvlabel, fwrap(#), fwrapon(style), cformat(%fmt), pformat(%fmt), sformat(%fmt), and nolstretch; see [R] estimation options.

Maximization

maximize_options: difficult, technique(algorithm_spec), iterate(#), [no]log, trace, gradient, showstep, hessian, showtolerance, tolerance(#), ltolerance(#), nrtolerance(#), nonrtolerance, and from(init_specs); see [R] maximize. These options are seldom used.
The following options are available with `probit` but are not shown in the dialog box:

- `nocoef` specifies that the coefficient table not be displayed. This option is sometimes used by programmers but is of no use interactively.
- `coeflegend`; see [R] estimation options.

### Remarks and examples

Remarks are presented under the following headings:

- Robust standard errors
- Model identification

`probit` fits maximum likelihood models with dichotomous dependent (left-hand-side) variables coded as 0/1 (more precisely, coded as 0 and not 0).

#### Example 1

We have data on the make, weight, and mileage rating of 22 foreign and 52 domestic automobiles. We wish to fit a probit model explaining whether a car is foreign based on its weight and mileage. Here is an overview of our data:

```
use http://www.stata-press.com/data/r13/auto
(1978 Automobile Data)
keep make mpg weight foreign
describe
Contains data from http://www.stata-press.com/data/r13/auto.dta
obs: 74 1978 Automobile Data
vars: 4 13 Apr 2013 17:45
size: 1,702 (_dta has notes)

+------------+------------------+
| variable   | storage type     | display format | label            |
|------------+------------------+----------------+------------------|
| make       | str18            | %-18s          | Make and Model   |
| mpg        | int              | %8.0g          | Mileage (mpg)    |
| weight     | int              | %8.0gc         | Weight (lbs.)    |
| foreign    | byte             | %8.0g          | origin           |

Sorted by: foreign
Note: dataset has changed since last saved
```

```
inspect foreign
Number of Observations

<table>
<thead>
<tr>
<th>Total</th>
<th>Integers</th>
<th>Nonintegers</th>
</tr>
</thead>
<tbody>
<tr>
<td>#</td>
<td>Negative</td>
<td>-</td>
</tr>
<tr>
<td>#</td>
<td>Zero</td>
<td>52</td>
</tr>
<tr>
<td>#</td>
<td>Positive</td>
<td>22</td>
</tr>
<tr>
<td>#    #</td>
<td>Total</td>
<td>74</td>
</tr>
<tr>
<td>#    #</td>
<td>Missing</td>
<td>-</td>
</tr>
</tbody>
</table>

(2 unique values)
foreign is labeled and all values are documented in the label.
```
The foreign variable takes on two unique values, 0 and 1. The value 0 denotes a domestic car, and 1 denotes a foreign car.

The model that we wish to fit is

\[
\Pr(\text{foreign} = 1) = \Phi(\beta_0 + \beta_1 \text{weight} + \beta_2 \text{mpg})
\]

where \(\Phi\) is the cumulative normal distribution.

To fit this model, we type

```
. probit foreign weight mpg
```

Iteration 0:  log likelihood =  -45.03321
Iteration 1:  log likelihood =  -27.914626
(output omitted)
Iteration 5:  log likelihood =  -26.844189

Probit regression
Number of obs = 74
LR chi2(2) = 36.38
Prob > chi2 = 0.0000
Log likelihood = -26.844189 Pseudo R2 = 0.4039

| foreign | Coef.  | Std. Err. | z     | P>|z|  | [95% Conf. Interval] |
|---------|--------|-----------|-------|------|----------------------|
| weight  | -0.0023355 | .0005661 | -4.13 | 0.000 | -.003445 -.0012261 |
| mpg     | -.1039503  | .0515689 | -2.02 | 0.044 | -.2050235 -.0028772 |
| _cons   | 8.275464   | 2.554142  | 3.24  | 0.001 | 3.269437 13.28149  |

We find that heavier cars are less likely to be foreign and that cars yielding better gas mileage are also less likely to be foreign, at least holding the weight of the car constant.

See [R] maximize for an explanation of the output.

⚠️ Technical note

Stata interprets a value of 0 as a negative outcome (failure) and treats all other values (except missing) as positive outcomes (successes). Thus if your dependent variable takes on the values 0 and 1, then 0 is interpreted as failure and 1 as success. If your dependent variable takes on the values 0, 1, and 2, then 0 is still interpreted as failure, but both 1 and 2 are treated as successes.

If you prefer a more formal mathematical statement, when you type `probit y x`, Stata fits the model

\[
\Pr(y_j \neq 0 | x_j) = \Phi(x_j \beta)
\]

where \(\Phi\) is the standard cumulative normal.

⚠️ Robust standard errors

If you specify the `vce(robust)` option, `probit` reports robust standard errors; see [U] 20.21 Obtaining robust variance estimates.

➤ Example 2

For the model from example 1, the robust calculation increases the standard error of the coefficient on mpg by almost 15%:
. probit foreign weight mpg, vce(robust) nolog

Probit regression
Number of obs = 74
Wald chi2(2) = 30.26
Prob > chi2 = 0.0000
Log pseudolikelihood = -26.844189  Pseudo R2 = 0.4039


<table>
<thead>
<tr>
<th></th>
<th>Robust</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>foreign</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>weight</td>
<td>-.0023355</td>
<td>.0004934</td>
<td>-4.73</td>
<td>0.000</td>
<td>-.0033025</td>
</tr>
<tr>
<td>mpg</td>
<td>-.1039503</td>
<td>.0593548</td>
<td>-1.75</td>
<td>0.080</td>
<td>-.2202836</td>
</tr>
<tr>
<td>_cons</td>
<td>8.275464</td>
<td>2.539177</td>
<td>3.26</td>
<td>0.001</td>
<td>3.298769</td>
</tr>
</tbody>
</table>

Without `vce(robust)`, the standard error for the coefficient on `mpg` was reported to be 0.052 with a resulting confidence interval of $[-0.21, 0.00]$.

Example 3

The `vce(cluster clustvar)` option can relax the independence assumption required by the probit estimator to independence between clusters. To demonstrate, we will switch to a different dataset.

We are studying unionization of women in the United States and have a dataset with 26,200 observations on 4,434 women between 1970 and 1988. We will use the variables `age` (the women were 14–26 in 1968, and our data span the age range of 16–46), `grade` (years of schooling completed, ranging from 0 to 18), `not_smsa` (28% of the person-time was spent living outside an SMSA—standard metropolitan statistical area), `south` (41% of the person-time was in the South), and `year`. Each of these variables is included in the regression as a covariate along with the interaction between `south` and `year`. This interaction, along with the `south` and `year` variables, is specified in the `probit` command using factor-variables notation, `south##c.year`. We also have variable `union`, indicating union membership. Overall, 22% of the person-time is marked as time under union membership, and 44% of these women have belonged to a union.
We fit the following model, ignoring that the women are observed an average of 5.9 times each in these data:

```
. use http://www.stata-press.com/data/r13/union, clear
    (NLS Women 14-24 in 1968)
. probit union age grade not_smsa south##c.year
Iteration 0:  log likelihood =  -13864.23
Iteration 1:  log likelihood =  -13545.541
Iteration 2:  log likelihood =  -13544.385
Iteration 3:  log likelihood =  -13544.385
Probit regression                           Number of obs =   26200
               LR chi2(6)  =      639.69
       Log likelihood =  -13544.385  Prob > chi2 = 0.0000
               Pseudo R2   =    0.0231

            union     Coef.  Std. Err.     z  P>|z|     [95% Conf. Interval]
    age   0.0118481    0.0029072     4.08 0.000     0.0061502    0.017546
    grade  0.0267365    0.0036689     7.29 0.000     0.0195457    0.0339273
    not_smsa -0.1293525    0.0202595    -6.38 0.000    -0.1690604   -0.0896445
     south  -0.8281077    0.2472219    -3.35 0.001    -1.312654   -0.3435618
     year  -0.0080931    0.0033469    -2.42 0.016    -0.0146529   -0.0015333
south#c.year  0.0057369    0.0030917     1.86 0.064    -0.0003226    0.0117965
    _cons -0.6542487    0.2007777    -3.26 0.001    -1.047766   -0.2607316
```

The reported standard errors in this model are probably meaningless. Women are observed repeatedly, and so the observations are not independent. Looking at the coefficients, we find a large southern effect against unionization and a time trend for the south that is almost significantly different from the overall downward trend. The vce(cluster clustvar) option provides a way to fit this model and obtains correct standard errors:
. probit union age grade not_smsa south##c.year, vce(cluster id)
Iteration 0:  log pseudolikelihood = -13864.23
Iteration 1:  log pseudolikelihood = -13545.541
Iteration 2:  log pseudolikelihood = -13544.385
Iteration 3:  log pseudolikelihood = -13544.385
Probit regression  Number of obs = 26200
Wald chi2(6) = 166.53
Prob > chi2 = 0.0000
Log pseudolikelihood = -13544.385  Pseudo R2 = 0.0231
(Std. Err. adjusted for 4434 clusters in idcode)
Robust

|        | Coef.     | Std. Err. | z    | P>|z| | [95% Conf. Interval] |
|--------|-----------|-----------|------|------|----------------------|
| union  |           |           |      |      |                      |
| age    | 0.0118481 | 0.0056625 | 2.09 | 0.036| 0.007499 0.0229463   |
| grade  | 0.0267365 | 0.0078124 | 3.42 | 0.001| 0.014244 0.0420486  |
| not_smsa | -.1293525 | 0.0403885 | -3.20| 0.001| -.2085125 -.0501925 |
| 1.south | -.8281077 | 0.3201584 | -2.59| 0.010| -.455607 -.206089   |
| year   | -.0080931 | 0.0060829 | -1.33| 0.183| -.0200153 .0038292  |
| south#c.year 1 | .0057369 | 0.0040133 | 1.43 | 0.153| -.002129 .0136029   |
| _cons  | -.6542487 | 0.3485976 | -1.88| 0.061| -1.337487 .02899    |

These standard errors are larger than those reported by the inappropriate conventional calculation. By comparison, another model we could fit is an equal-correlation population-averaged probit model:

. xtprobit union age grade not_smsa south##c.year, pa
Iteration 1:  tolerance = .12544249
Iteration 2:  tolerance = .0034686
Iteration 3:  tolerance = .00017448
Iteration 4:  tolerance = 8.382e-06
Iteration 5:  tolerance = 3.997e-07
GEE population-averaged model  Number of obs = 26200
Group variable: idcode Number of groups = 4434
Link: probit Obs per group: min = 1
Family: binomial avg = 5.9
Correlation: exchangeable max = 12
Scale parameter: 1 Wald chi2(6) = 242.57
Prob > chi2 = 0.0000

|        | Coef.     | Std. Err. | z    | P>|z| | [95% Conf. Interval] |
|--------|-----------|-----------|------|------|----------------------|
| union  |           |           |      |      |                      |
| age    | 0.0089699 | 0.0053208 | 1.69 | 0.092| -.0014586 .0193985  |
| grade  | 0.0333174 | 0.0062352 | 5.34 | 0.000| 0.0210966 0.045382  |
| not_smsa | -.0715717 | 0.027543  | -2.60| 0.009| -.1255551 -.0175884 |
| 1.south | -1.017368 | 0.207931  | -4.89| 0.000| -1.424905 -.6098308 |
| year   | -.0062708 | 0.0055314 | -1.13| 0.257| -.0171122 .0045706  |
| south#c.year 1 | .0086294 | 0.00258  | 3.34 | 0.001| .0035727 .013686   |
| _cons  | -.8670997 | 0.294771  | -2.94| 0.003| -1.44484 -.2893592  |
The coefficient estimates are similar, but these standard errors are smaller than those produced by `probit, vce(cluster clustvar)`, as we would expect. If the equal-correlation assumption is valid, the population-averaged probit estimator above should be more efficient.

Is the assumption valid? That is a difficult question to answer. The default population-averaged estimates correspond to an assumption of exchangeable correlation within person. It would not be unreasonable to assume an AR(1) correlation within person or to assume that the observations are correlated but that we do not wish to impose any structure. See [XT] `xtprobit` and [XT] `xtgee` for full details.

`probit, vce(cluster clustvar)` is robust to assumptions about within-cluster correlation. That is, it inefficiently sums within cluster for the standard error calculation rather than attempting to exploit what might be assumed about the within-cluster correlation.

**Model identification**

The `probit` command has one more feature that is probably the most useful. It will automatically check the model for identification and, if the model is underidentified, drop whatever variables and observations are necessary for estimation to proceed.

> Example 4

Have you ever fit a probit model where one or more of your independent variables perfectly predicted one or the other outcome?

For instance, consider the following data:

<table>
<thead>
<tr>
<th>Outcome y</th>
<th>Independent variable x</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Say that we wish to predict the outcome on the basis of the independent variable. The outcome is always zero when the independent variable is one. In our data, \( \Pr(y = 0 \mid x = 1) = 1 \), which means that the probit coefficient on \( x \) must be minus infinity with a corresponding infinite standard error. At this point, you may suspect that we have a problem.

Unfortunately, not all such problems are so easily detected, especially if you have many independent variables in your model. If you have ever had such difficulties, then you have experienced one of the more unpleasant aspects of computer optimization. The computer has no idea that it is trying to solve for an infinite coefficient as it begins its iterative process. All it knows is that, at each step, making the coefficient a little bigger, or a little smaller, works wonders. It continues on its merry way until either 1) the whole thing comes crashing to the ground when a numerical overflow error occurs or 2) it reaches some predetermined cutoff that stops the process. Meanwhile, you have been waiting. And the estimates that you finally receive, if any, may be nothing more than numerical roundoff.

Stata watches for these sorts of problems, alerts you, fixes them, and then properly fits the model.

Let’s return to our automobile data. Among the variables we have in the data is one called `repair` that takes on three values. A value of 1 indicates that the car has a poor repair record, 2 indicates an average record, and 3 indicates a better-than-average record. Here is a tabulation of our data:
. use http://www.stata-press.com/data/r13/repair
(1978 Automobile Data)
. tabulate foreign repair

<table>
<thead>
<tr>
<th>Car type</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domestic</td>
<td>10</td>
<td>27</td>
<td>9</td>
<td>46</td>
</tr>
<tr>
<td>Foreign</td>
<td>0</td>
<td>3</td>
<td>9</td>
<td>12</td>
</tr>
<tr>
<td>Total</td>
<td>10</td>
<td>30</td>
<td>18</td>
<td>58</td>
</tr>
</tbody>
</table>

All the cars with poor repair records (repair = 1) are domestic. If we were to attempt to predict foreign on the basis of the repair records, the predicted probability for the repair = 1 category would have to be zero. This in turn means that the probit coefficient must be minus infinity, and that would set most computer programs buzzing.

Let’s try using Stata on this problem.

. probit foreign b3.repair

note: 1.repair != 0 predicts failure perfectly
1.repair dropped and 10 obs not used

Iteration 0: log likelihood = -26.992087
Iteration 1: log likelihood = -22.276479
Iteration 2: log likelihood = -22.229184
Iteration 3: log likelihood = -22.229138
Iteration 4: log likelihood = -22.229138

Probit regression

| Coef.  | Std. Err. | z    | P>|z| | [95% Conf. Interval] |
|--------|-----------|------|------|---------------------|
| repair |           |      |      |                     |
| 1      | 0         |      |      |                     |
| 2      | -1.281552 | .4297326 | -2.98 | 0.003 | -2.123812 | -.4392911 |
| _cons  | 9.89e-17  | .295409 | 0.00 | 1.000 | -.578991 | .578991 |

Remember that all the cars with poor repair records (repair = 1) are domestic, so the model cannot be fit, or at least it cannot be fit if we restrict ourselves to finite coefficients. Stata noted that fact “note: 1.repair != 0 predicts failure perfectly”. This is Stata’s mathematically precise way of saying what we said in English. When repair is 1, the car is domestic.

Stata then went on to say, “1.repair dropped and 10 obs not used”. This is Stata eliminating the problem. First, 1.repair had to be removed from the model because it would have an infinite coefficient. Then the 10 observations that led to the problem had to be eliminated, as well, so as not to bias the remaining coefficients in the model. The 10 observations that are not used are the 10 domestic cars that have poor repair records.

Stata then fit what was left of the model, using the remaining observations. Because no observations remained for cars with poor repair records, Stata reports “(empty)” in the row for repair = 1.
Technical note

Stata is pretty smart about catching these problems. It will catch “one-way causation by a dummy variable”, as we demonstrated above.

Stata also watches for “two-way causation”, that is, a variable that perfectly determines the outcome, both successes and failures. Here Stata says that the variable “predicts outcome perfectly” and stops. Statistics dictate that no model can be fit.

Stata also checks your data for collinear variables; it will say “so-and-so omitted because of collinearity”. No observations need to be eliminated here and model fitting will proceed without the offending variable.

It will also catch a subtle problem that can arise with continuous data. For instance, if we were estimating the chances of surviving the first year after an operation, and if we included in our model age, and if all the persons over 65 died within the year, Stata will say, “age > 65 predicts failure perfectly”. It will then inform us about how it resolves the issue and fit what can be fit of our model.

`probit` (and `logit`, `logistic`, and `ivprobit`) will also occasionally fail to converge and then display messages such as

```
Note: 4 failures and 0 successes completely determined.
```

The cause of this message and what to do if you see it are described in [R] `logit`.

Stored results

`probit` stores the following in `e()`:

Scalars

- `e(N)` number of observations
- `e(N_cds)` number of completely determined successes
- `e(N_cdf)` number of completely determined failures
- `e(k)` number of parameters
- `e(k_eq)` number of equations in `e(b)`
- `e(k_eq_model)` number of equations in overall model test
- `e(k_eq_model)` number of dependent variables
- `e(df_m)` model degrees of freedom
- `e(r2_p)` pseudo-$R$-squared
- `e(ll)` log likelihood
- `e(ll_0)` log likelihood, constant-only model
- `e(N_cluster)` number of clusters
- `e(chi2)` $\chi^2$
- `e(p)` significance of model test
- `e(rank)` rank of `e(V)`
- `e(ic)` number of iterations
- `e(rc)` return code
- `e(converged)` 1 if converged, 0 otherwise
Macros

- `e(cmd)` probit
- `e(cmdline)` command as typed
- `e(depvar)` name of dependent variable
- `e(wtype)` weight type
- `e(wexp)` weight expression
- `e(title)` title in estimation output
- `e(clustvar)` name of cluster variable
- `e(offset)` linear offset variable
- `e(chi2type)` Wald or LR; type of model \( \chi^2 \) test
- `e(vce)` vcetype specified in `vce()`
- `e(vcetype)` title used to label Std. Err.
- `e(opt)` type of optimization
- `e(which)` max or min; whether optimizer is to perform maximization or minimization
- `e(ml_method)` type of ml method
- `e(user)` name of likelihood-evaluator program
- `e(technique)` maximization technique
- `e(properties)` b V
- `e(estat_cmd)` program used to implement estat
- `e(predict)` program used to implement predict
- `e(asbalanced)` factor variables fvset as asbalanced
- `e(asobserved)` factor variables fvset as asobserved

Matrices

- `e(b)` coefficient vector
- `e(Cns)` constraints matrix
- `e(ilog)` iteration log (up to 20 iterations)
- `e(gradient)` gradient vector
- `e(mns)` vector of means of the independent variables
- `e(rules)` information about perfect predictors
- `e(V)` variance–covariance matrix of the estimators
- `e(V_modelbased)` model-based variance

Functions

- `e(sample)` marks estimation sample

Methods and formulas

Probit analysis originated in connection with bioassay, and the word probit, a contraction of “probability unit”, was suggested by Bliss (1934a, 1934b). For an introduction to probit and logit, see, for example, Aldrich and Nelson (1984), Cameron and Trivedi (2010), Greene (2012), Long (1997), Pampel (2000), or Powers and Xie (2008). Long and Freese (2014, chap. 5 and 6) and Jones (2007, chap. 3) provide introductions to probit and logit, along with Stata examples.

The log-likelihood function for probit is

\[
\ln L = \sum_{j \in S} w_j \ln \Phi(x_j \beta) + \sum_{j \notin S} w_j \ln \left\{ 1 - \Phi(x_j \beta) \right\}
\]

where \( \Phi \) is the cumulative normal and \( w_j \) denotes the optional weights. \( \ln L \) is maximized, as described in \([R]\) maximize.

This command supports the Huber/White/sandwich estimator of the variance and its clustered version using `vce(robust)` and `vce(cluster clustvar)`, respectively. See \([P]\) _robust_, particularly Maximum likelihood estimators and Methods and formulas. The scores are calculated as \( u_j = \{ \phi(x_j \beta) / \Phi(x_j \beta) \} x_j \) for the positive outcomes and \(-\{ \phi(x_j \beta) / (1 - \Phi(x_j \beta)) \} x_j \) for the negative outcomes, where \( \phi \) is the normal density.

probit also supports estimation with survey data. For details on VCEs with survey data, see \([SVY]\) variance estimation.
Chester Ittner Bliss (1899–1979) was born in Ohio. He was educated as an entomologist, earning degrees from Ohio State and Columbia, and was employed by the United States Department of Agriculture until 1933. When he lost his job because of the Depression, Bliss then worked with R. A. Fisher in London and at the Institute of Plant Protection in Leningrad before returning to a post at the Connecticut Agricultural Experiment Station in 1938. He was also a lecturer at Yale for 25 years. Among many contributions to biostatistics, his development and application of probit methods to biological problems are outstanding.

References


Also see

[R] probit postestimation — Postestimation tools for probit
[R] asmprobit — Alternative-specific multinomial probit regression
[R] biprobit — Bivariate probit regression
[R] brier — Brier score decomposition
[R] glm — Generalized linear models
[R] heckoprobit — Ordered probit model with sample selection
[R] hetprobit — Heteroskedastic probit model
[R] ivprobit — Probit model with continuous endogenous regressors
[R] logistic — Logistic regression, reporting odds ratios
[R] logit — Logistic regression, reporting coefficients
[R] mprobit — Multinomial probit regression
[R] roc — Receiver operating characteristic (ROC) analysis
[R] scobit — Skewed logistic regression
[ME] meprobit — Multilevel mixed-effects probit regression
[M] estimation — Estimation commands for use with mi estimate
[SVY] svy estimation — Estimation commands for survey data
[XT] xtprobit — Random-effects and population-averaged probit models
[U] 20 Estimation and postestimation commands