27 Overview of Stata estimation commands

Contents

27.1 Introduction

Stata has many estimation commands that compute summary statistics and fit statistical models, so it is easy to overlook a few. Some of these commands differ greatly from each other, others are gentle variations on a theme, and still others are equivalent to each other.

There are also estimation prefixes that modify the calculation performed by the command, such as `svy:`, `mi:`, `bayes:`, and `fmm:`.

The majority of Stata's estimation commands share features that this chapter will not discuss; see **[U] 20 Estimation and postestimation commands**. Especially see **[U] 20.22 Obtaining robust variance estimates**, which discusses an alternative calculation for the estimated variance matrix (and hence standard errors) that many of Stata's estimation commands provide. Also see **[U] 20.13 Performing hypothesis tests on the coefficients**. This overview chapter, however, will put all that aside and deal solely with matching commands to their statistical concepts.

This chapter discusses all the official estimation commands included in Stata 19. Users may have written their own estimation commands that they are willing to share. Type `search estimation`, `ssc new`, or `ssc hot` to discover more estimation commands; see **[R] ssc**. And, of course, you can always write your own commands.

27.2 Means, proportions, and related statistics

This group of estimation commands computes summary statistics rather than fitting regression models. However, being estimation commands, they share the features discussed in **[U] 20 Estimation and postestimation commands**, such as allowing the use of postestimation commands.

`mean`, `proportion`, `ratio`, and `total` provide estimates of population means, proportions, ratios, and totals, respectively. Each of these commands allows for obtaining separate estimates within subpopulations, groups defined by a separate categorical variable. In addition, `mean`, `proportion`, and `ratio` can report statistics adjusted by direct standardization.

pwmean provides another option for computing means of one variable for each level of one or more categorical variables. In addition, pwmean computes all pairwise differences in these means along with the corresponding tests and confidence intervals (CIs), which can optionally be adjusted to account for multiple comparisons.

27.3 Continuous outcomes

27.3.1 ANOVA and ANCOVA

ANOVA and ANCOVA fit general linear models and are related to the linear regression models discussed in **[U] 27.3.2 Linear regression**, but we classify them separately. The related Stata commands are anova, oneway, and loneway.

anova fits ANOVA and ANCOVA models, one-way and up—including two-way factorial, three-way factorial, etc.—and fits nested and mixed-design models as well as repeated-measures models.

oneway fits one-way ANOVA models. It reports Bartlett's test for equal variance and can also report multiple-comparison tests. After anova, use pwcompare to perform multiple-comparison tests.

loneway is an alternative to oneway. The results are numerically the same, but loneway can deal with more levels, limited only by dataset size (oneway is limited to 376 levels). loneway also reports some additional statistics, such as the intraclass correlation.

For MANOVA and MANCOVA, see **[U] 27.22 Multivariate analysis**.

27.3.2 Linear regression

Consider models of the form

$$y_j = \mathbf{x}_j\boldsymbol{\beta} + \epsilon_j$$

for a continuous y variable and where σ^2_ϵ is constant across observations j. The model is called the linear regression model, and the estimator is often called the (ordinary) least-squares (OLS) estimator.

regress is Stata's linear regression command. regress produces the robust estimate of variance as well as the conventional estimate, and regress has a collection of commands that can be run after it to explore the nature of the fit.

The following commands will also do linear regressions, but they offer special features:

1. areg fits models $y_j = \mathbf{x}_j\boldsymbol{\beta} + \mathbf{d}_j\boldsymbol{\gamma} + \epsilon_j$, where $\mathbf{d}_j$ is a mutually exclusive and exhaustive dummy variable set. areg obtains estimates of $\boldsymbol{\beta}$ (and associated statistics) without ever forming $\mathbf{d}_j$, meaning that it also does not report the estimated $\boldsymbol{\gamma}$. If your interest is in fitting fixed-effects models, Stata has a better command—xtreg—discussed in **[U] 27.15.1 Continuous outcomes with panel data**. Most users who find areg appealing will probably want to use xtreg because it provides more useful summary and test statistics. areg duplicates the output that regress would produce if you were to generate all the dummy variables. This means, for instance, that the reported R^2 includes the effect of $\boldsymbol{\gamma}$.
2. wildbootstrap regress and wildbootstrap areg fit the same models as regress and areg but provides wild cluster bootstrap p-values and CIs for robust inference. See **[R] wildbootstrap** for details.
3. cnsreg allows you to place linear constraints on the coefficients.

4. eivreg adjusts estimates for errors in variables.
5. rreg fits robust regression models, which are not to be confused with regression with robust standard errors. Robust standard errors are discussed in [U] 20.22 Obtaining robust variance estimates. Robust regression concerns point estimates more than it does standard errors, and it implements a data-dependent method for downweighting outliers.

27.3.3 Regression with heteroskedastic errors

We now consider the model $y_j = \mathbf{x}_j\boldsymbol{\beta} + \epsilon_j$, where the variance of ϵ_j is nonconstant.

hetregress fits models with multiplicative heteroskedasticity, that is, models in which the variance of ϵ_j is an exponential function of one or more covariates. The heteroskedasticity can be modeled using either maximum likelihood or Harvey's two-step generalized least-squares method.

When not much is known about the functional form of the variance of ϵ_j, regress with the vce(robust) option is preferred because it provides unbiased estimates. What Stata calls robust is also known as the White correction for heteroskedasticity.

vwls (variance-weighted least squares) produces estimates of $y_j = \mathbf{x}_j\boldsymbol{\beta}+\epsilon_j$, where the variance of ϵ_j is calculated from group data or is known a priori. vwls is therefore of most interest to categorical-data analysts and physical scientists.

qreg performs quantile regression, which in the presence of heteroskedasticity is most of interest. Median regression (one of qreg's capabilities) is an estimator of $y_j = \mathbf{x}_j\boldsymbol{\beta}+\epsilon_j$ when ϵ_j is heteroskedastic. Even more useful, you can fit models of other quantiles and so model the heteroskedasticity. bsqreg is identical to qreg but reports bootstrap standard errors. Also see the sqreg and iqreg commands; sqreg estimates multiple quantiles simultaneously, and iqreg estimates differences in quantiles.

27.3.4 Estimation with correlated errors

By correlated errors, we mean that observations are grouped; within a group, the observations might be correlated, but across groups, they are uncorrelated. regress with the vce(cluster *clustvar*) option can produce "correct" estimates, that is, inefficient estimates with correct standard errors and a lot of robustness; see [U] 20.22 Obtaining robust variance estimates. Alternatively, you can model the within-group correlation using xtreg, xtgls, or mixed; we discuss these commands in [U] 27.15.1 Continuous outcomes with panel data and [U] 27.16 Multilevel mixed-effects models.

Estimation in the presence of autocorrelated errors is discussed in [U] 27.14 Time-series models.

27.3.5 Regression with censored or truncated outcomes

1. tobit allows estimation of linear regression models when y_i has been subject to left-censoring, right-censoring, or both. Say that y_i is not observed if $y_i < 1000$, but for those observations, it is known that $y_i < 1000$. tobit fits such models.
2. intreg (interval regression) is a generalization of tobit. In addition to allowing open-ended intervals, intreg allows closed intervals. Rather than observing y_j, intreg assumes that y_{0j} and y_{1j} are observed, where $y_{0j} \leq y_j \leq y_{1j}$. Survey data might report that a subject's monthly income was in the range \$1,500–\$2,500. intreg allows such data to be used to fit a regression model. intreg allows $y_{0j} = y_{1j}$ and so can reproduce results reported by regress. intreg allows y_{0j} to be $-\infty$ and y_{1j} to be $+\infty$ and so can reproduce results reported by tobit.

3. truncreg fits the regression model when the sample is drawn from a restricted part of the population and so is similar to tobit, except that here the independent variables are not observed. Under the normality assumption for the whole population, the error terms in the truncated regression model have a truncated-normal distribution.

4. churdle allows estimation of linear or exponential hurdle models when y_i is subject to a lower boundary $\ell\ell$, an upper boundary $u\ell$, or both. The dependent variable is a mixture of discrete observations at the boundary points and continuous observations over the interior. Both boundary and interior observations on y_i are actual realizations. In contrast, censored-data models such as tobit and intreg treat interior observations as actual realizations and treat boundary observations as indicating only that the actual realizations lie beyond the boundary. Hurdle models use one model to determine whether an observation is on the boundary or in the interior and another model for the values in the interior.

27.3.6 Multiple-equation models

When we have errors that are correlated across equations but not correlated with any of the right-hand-side variables, we can write the system of equations as

$$\begin{aligned} y_{1j} &= \mathbf{x}_{1j}\boldsymbol{\beta} + \epsilon_{1j} \\ y_{2j} &= \mathbf{x}_{2j}\boldsymbol{\beta} + \epsilon_{2j} \\ &\vdots \\ y_{mj} &= \mathbf{x}_{mj}\boldsymbol{\beta} + \epsilon_{mj} \end{aligned}$$

where $\epsilon_{k\cdot}$ and $\epsilon_{l\cdot}$ are correlated with correlation ρ_{kl}, a quantity to be estimated from the data. This is called Zellner's seemingly unrelated regression, and sureg fits such models. When $\mathbf{x}_{1j} = \mathbf{x}_{2j} = \cdots = \mathbf{x}_{mj}$, the model is known as multivariate regression, and the corresponding command is mvreg.

The equations need not be linear. If they are not linear, use nlsur; see **[U] 27.3.8 Nonlinear regression**. Also, see demandsys for fitting a set of equations that describe consumers' purchases of goods or services.

27.3.7 Stochastic frontier models

frontier fits stochastic production or cost frontier models on cross-sectional data. The model can be expressed as

$$y_i = \mathbf{x}_i\boldsymbol{\beta} + v_i - su_i$$

where

$$s = \begin{cases} 1 & \text{for production functions} \\ -1 & \text{for cost functions} \end{cases}$$

u_i is a nonnegative disturbance standing for technical inefficiency in the production function or cost inefficiency in the cost function. Although the idiosyncratic error term v_i is assumed to have a normal distribution, the inefficiency term is assumed to be one of the three distributions: half-normal, exponential, or truncated-normal. Also, when the nonnegative component of the disturbance is assumed to be either half-normal or exponential, frontier can fit models in which the error components are heteroskedastic conditional on a set of covariates. When the nonnegative component of the disturbance is assumed to be from a truncated-normal distribution, frontier can also fit a conditional mean model, where the mean of the truncated-normal distribution is modeled as a linear function of a set of covariates.

For panel-data stochastic frontier models, see **[U] 27.15.1 Continuous outcomes with panel data**.

27.3.8 Nonlinear regression

`nl` provides the nonlinear least-squares estimator of $y_j = f(\mathbf{x}_j, \boldsymbol{\beta}) + \epsilon_j$, where $f(\mathbf{x}_j, \boldsymbol{\beta})$ is an arbitrary nonlinear regression function such as the exponential or the lognormal. `nlsur` fits a system of nonlinear equations by feasible generalized nonlinear least squares. It can be viewed as a nonlinear variant of Zellner's seemingly unrelated regression model.

A special case of a nonlinear model is the Box–Cox transform. `boxcox` obtains maximum likelihood estimates of the coefficients and the Box–Cox transform parameters in a model of the form

$$y_i^{(\theta)} = \beta_0 + \beta_1 x_{i1}^{(\lambda)} + \beta_2 x_{i2}^{(\lambda)} + \cdots + \beta_k x_{ik}^{(\lambda)} + \gamma_1 z_{i1} + \gamma_2 z_{i2} + \cdots + \gamma_l z_{il} + \epsilon_i$$

where $\epsilon \sim N(0, \sigma^2)$. Here the y is subject to a Box–Cox transform with parameter θ. Each of the $x_1, x_2, \ldots, x_k$ is transformed by a Box–Cox transform with parameter λ. The $z_1, z_2, \ldots, z_l$ are independent variables that are not transformed. In addition to the general form specified above, `boxcox` can fit three other versions of this model defined by the restrictions $\lambda = \theta$, $\lambda = 1$, and $\theta = 1$.

For nonlinear mixed-effects models, see **[U] 27.16 Multilevel mixed-effects models**.

27.3.9 Nonparametric regression

All the models discussed so far have specified a particular functional form for the relationship between the outcome and the covariates in the model. Nonparametric regression allows you to model the mean of an outcome given the covariates when you are uncertain about its functional form. Stata's commands for nonparametric estimation are `npregress kernel` and `npregress series`. In general, for any outcome that you would be comfortable modeling using `regress`, you can use `npregress kernel` or `npregress series`. The difference is that you no longer have to assume a linear relationship. However, you need more observations for nonparametric estimators than you need for the parametric estimators.

`npregress kernel` implements two nonparametric kernel-based estimators—a local-linear estimator and a local-constant estimator. These kernel-based estimators rely on finding an optimal bandwidth parameter that balances the tradeoff between bias and variance. Both of these kernel-based estimators provide equivalent estimators of the mean, but there are some important differences to consider. The local-linear estimator lets you obtain marginal effects for continuous covariates. Also, if the model is linear, the local-linear estimator will recover a linear mean, whereas the local constant may not. For cases in which your outcome is nonnegative, the local-constant estimator will yield nonnegative predictions. The local-linear estimator may result in negative predictions in such cases.

`npregress series` implements nonparametric series estimators using a B-spline, spline, or polynomial basis. In series estimation, the unknown mean function is approximated by a linear combination of elements in the basis function. For instance, we can consider a polynomial basis. We can approximate the unknown mean function using a polynomial. The more complex the mean function, the more polynomial terms (`x`, $\texttt{x}^2$, $\texttt{x}^3$, ...) we need to include to approximate the mean consistently. Likewise, as the complexity of the the mean function increases, we need more terms in a B-spline or spline basis function. `npregress series` selects the number of terms that optimally balances the tradeoff between bias and variance. Once the terms are selected, we fit a least-squares regression. Having a linear representation of the approximating function and using it to construct inferences makes series estimation appealing.

See [R] **npregress intro** for more information about nonparametric regression.

27.4 Binary outcomes

27.4.1 Logistic, probit, and complementary log–log regression

There are many ways to write these models, one of which is

$$\Pr(y_j \neq 0) = F(\mathbf{x}_j\boldsymbol{\beta})$$

where F is some cumulative distribution. Two popular choices for $F(\cdot)$ are the normal and logistic, and the models are called the probit and logit (or logistic regression) models. The two parent commands for the maximum likelihood estimator of probit and logit are `probit` and `logit`. `logit` has a sibling, `logistic`, that provides the same estimates but displays results in a slightly different way. There is also an exact logistic estimator; see **[U] 27.11 Exact estimators**.

A third choice for $F(\cdot)$ is the complementary log–log function. Maximum likelihood estimates are obtained by Stata's `cloglog` command.

Do not read anything into the names logit and logistic. Logit and logistic have two interchanged definitions in two scientific camps. In the medical sciences, logit means the minimum χ^2 estimator, and logistic means maximum likelihood. In the social sciences, it is the other way around. From our experience, it appears that neither reads the other's literature, because both assert that logit means one thing and logistic the other. Our solution is to provide both `logit` and `logistic`, which do the same thing, so that each camp can latch on to the maximum likelihood command under the name each expects.

There are two slight differences between `logit` and `logistic`. `logit` reports estimates in the coefficient metric, whereas `logistic` reports exponentiated coefficients—odds ratios. This is in accordance with the expectations of each camp and makes no substantial difference. The other difference is that `logistic` has a family of post-`logistic` commands that you can run to explore the nature of the fit. Actually, that is not exactly true because all the commands for use after `logistic` can also be used after `logit`.

If you have not already selected `logit` or `logistic` as your favorite, we recommend that you try `logistic`. Logistic regression (logit) models are more easily interpreted in the odds-ratio metric.

`binreg` can be used to model either individual-level or grouped data in an application of the generalized linear model. The family is assumed to be binomial, and each link provides a distinct parameter interpretation. Also, `binreg` offers several options for setting the link function according to the desired biostatistical interpretation. The available links and interpretation options are the following:

Option	Implied link	Parameter
`or`	`logit`	Odds ratios = exp(β)
`rr`	`log`	Risk ratios = exp(β)
`hr`	`log complement`	Health ratios = exp(β)
`rd`	`identity`	Risk differences = β

`reri` can be used to assess two-way interactions in an additive model of relative risk. It reports the relative excess risk due to interaction, attributable proportion, and synergy index. `reri` supports binomial generalized linear and logistic models, as well as Poisson, negative binomial, Cox, parametric survival, and interval-censored parametric and nonparametric survival models.

Related to logit, the skewed logit estimator scobit adds a power to the logit link function and is estimated by Stata's `scobit` command.

`hetprobit` fits heteroskedastic probit models. In these models, the variance of the error term is parameterized.

Also, Stata's biprobit command fits bivariate probit models, meaning models with two correlated outcomes. biprobit also fits partial-observability models in which only the outcomes $(0,0)$ and $(1,1)$ are observed.

27.4.2 Conditional logistic regression

clogit is Stata's conditional logistic regression estimator. In this model, observations are assumed to be partitioned into groups, and a predetermined number of events occur in each group. The model measures the risk of the event according to the observation's covariates, $\mathbf{x}_j$. The model is used in matched case–control studies (clogit allows $1:1$, $1:k$, and $m:k$ matching) and is used in natural experiments whenever observations can be grouped into pools in which a fixed number of events occur. clogit is also used to fit logistic regression with fixed group effects.

27.4.3 ROC analysis

ROC stands for "receiver operating characteristics". ROC deals with specificity and sensitivity, the number of false positives and undetected true positives of a diagnostic test. The term "ROC" dates back to the early days of radar when there was a knob on the radar receiver labeled "ROC". If you turned the knob one way, the receiver became more sensitive, which meant it was more likely to show airplanes that really were there and, simultaneously, more likely to show returns where there were no airplanes (false positives). If you turned the knob the other way, you suppressed many of the false positives, but unfortunately, you also suppressed the weak returns from real airplanes (undetected positives). These days, in the statistical applications we imagine, one does not turn a knob but instead chooses a value of the diagnostic test, above which is declared to be a positive and below which, a negative.

ROC analysis is applied to binary outcomes such as those appropriate for probit or logistic regression. After fitting a model, one can obtain predicted probabilities of a positive outcome. One chooses a value, above which the predicted probability is declared a positive and below which, a negative.

ROC analysis is about modeling the tradeoff of sensitivity and specificity as the threshold value is chosen.

Stata's suite for ROC analysis consists of six commands: roctab, roccomp, rocfit, rocgold, rocreg, and rocregplot.

roctab provides nonparametric estimation of the ROC curve and produces Bamber and Hanley CIs for the area under the curve.

roccomp provides tests of equality of ROC areas. It can estimate nonparametric and parametric binormal ROC curves.

rocfit fits maximum likelihood models for a single classifier, an indicator of the latent binormal variable for the true status.

rocgold performs tests of equality of ROC areas against a "gold standard" ROC curve and can adjust significance levels for multiple tests across classifiers via Šidák's method.

rocreg performs ROC regression; it can adjust both sensitivity and specificity for prognostic factors such as age and gender. It is by far the most general of all the ROC commands.

rocregplot graphs ROC curves as modeled by rocreg. ROC curves can be drawn across covariate values, across classifiers, and across both.

See [R] **roc**.

27.5 Fractional outcomes

Fractional response data occur when the outcome of interest is measured as a fraction, proportion, or rate. Two widely used methods for modeling these outcomes are beta regression and fractional regression.

betareg can be used to estimate the parameters of a beta-regression model for fractional responses that are strictly greater than zero and less than one.

fracreg can be used to estimate the parameters of a fractional logistic model, a fractional probit model, or a fractional heteroskedastic probit model for fractional responses that are greater than or equal to zero and less than or equal to one.

Both commands use quasimaximum likelihood estimation. When the dependent variable is between zero and one, betareg provides more flexibility than fracreg in the distribution of the conditional mean of the dependent variable.

27.6 Ordinal outcomes

For ordered outcomes, Stata provides ordered logit, ordered probit, zero-inflated ordered probit, and rank-ordered logit, as well as rank-ordered logit regression.

oprobit and ologit provide maximum-likelihood ordered probit and logit. These are generalizations of probit and logit models known as the proportional odds model and are used when the outcomes have a natural ordering from low to high. The idea is that there is an unmeasured $z_j = \mathbf{x}_j\boldsymbol{\beta}$, and the probability that the kth outcome is observed is $\Pr(c_{k-1} < z_j < c_k)$, where $c_0 = -\infty$, $c_k = +\infty$, and $c_1, \ldots, c_{k-1}$ along with $\boldsymbol{\beta}$ are estimated from the data.

hetoprobit fits heteroskedastic ordered probit models to ordinal outcomes. It is a generalization of an ordered probit model that allows the variance to be modeled as a function of independent variables and to differ between subjects or population groups.

zioprobit fits zero-inflated ordered probit models, and ziologit fits zero-inflated ordered logit models. They are used to model an ordered outcome with a higher fraction of observations in the lowest category than would be expected from an ordered probit or ordered logit model. Typically, the lowest value is represented by zero—hence the name, "zero inflated". In these models, the outcome is a result of two processes. In the zero-inflated ordered probit model, a probit process first describes the presence of excess zeros or "nonparticipants". Second, an ordered probit process, conditional on "participation" from the probit process, describes the ordered outcome. Zero-inflated ordered logit is similar, except excess zeros are modeled with a logit process and an ordered logit process describes the ordered outcome. The logit version can optionally report odds ratios.

cmrologit fits the rank-ordered logit model for rankings. This model is also known as the Plackett–Luce model, the exploded logit model, and choice-based conjoint analysis.

cmroprobit fits the probit model for rankings, a more flexible estimator than cmrologit because cmroprobit allows covariances among the rankings.

27.7 Categorical outcomes

For categorical outcomes, Stata provides multinomial logistic regression, multinomial probit regression, stereotype logit regression, McFadden's choice model (conditional fixed-effects logistic regression), nested logistic regression, multinomial probit choice model, and mixed logit choice model.

mlogit fits maximum-likelihood multinomial logistic models, also known as polytomous logistic regression. mprobit is similar but instead is a generalization of the probit model. Both models are intended for use when the outcomes have no natural ordering and you know only the characteristics of the outcome chosen (and, perhaps, the chooser).

slogit fits the stereotype logit model for data that are not truly ordered, as data are for ologit, but for which you are not sure that they are unordered, in which case mlogit would be appropriate.

cmclogit fits McFadden's choice model, also known as conditional logistic regression. In the context denoted by the name McFadden's choice model, the model is used when the outcomes have no natural ordering, just as in multinomial logistic regression, but the characteristics of the outcomes chosen and not chosen are known (along with, perhaps, the characteristics of the chooser).

In the context denoted by the name conditional logistic regression—mentioned above—subjects are members of pools, and one or more are chosen, typically to be infected by some disease or to have some other unfortunate event befall them. Thus, the characteristics of the chosen and not chosen are known, and the issue of the characteristics of the chooser never arises. Either way, it is the same model.

In their choice-model interpretations, mlogit and cmclogit assume that the odds ratios are independent of other alternatives, known as the independence of irrelevant alternatives (IIA) assumption. This assumption is often rejected by the data, and the nested logit model relaxes this assumption. nlogit is also popular for fitting the random utility choice model.

cmmprobit is for use with outcomes that have no natural ordering and with regressors that are alternative specific. cmmixlogit is a generalization of mlogit that allows for correlation of choices across outcomes. Unlike mlogit, cmmprobit and cmmixlogit do not assume the IIA. The random coefficients that are used by cmmixlogit to relax the IIA also directly model the heterogeneity in choices given covariates.

27.8 Count outcomes

These models concern dependent variables that count the occurrences of an event. In this category, we include Poisson and negative binomial regression. For the Poisson model,

$$E(\text{count}) = E_j \exp(\mathbf{x}_j\boldsymbol{\beta})$$

where E_j is the exposure time. poisson fits this model. There is also an exact Poisson estimator; see **[U] 27.11 Exact estimators**.

Negative binomial regression refers to estimating with data that are a mixture of Poisson counts. One derivation of the negative binomial model is that individual units follow a Poisson regression model but that there is an omitted variable that follows a gamma distribution with parameter α. Negative binomial regression estimates $\boldsymbol{\beta}$ and α. nbreg fits such models. A variation on this, unique to Stata, allows you to model α. gnbreg fits those models.

Sometimes, the value of the outcome variable is not observed when it falls outside a known range, and it is observed inside that range. This limitation comes in two types—censoring and truncation. It is called censoring when we have an observation for the outcome but know only that the value of the outcome is outside the range. It is called truncation when we do not even have an observation when the value of the outcome is outside the range. The cpoisson command can be used to fit models for censored count data. Commands tpoisson and tnbreg can be used to fit models for truncated count data.

Zero inflation refers to count models in which the number of zero counts is more than would be expected in the regular model. The excess zeros are explained by a preliminary probit or logit process. If the preliminary process produces a positive outcome, the usual counting process occurs, and otherwise, the count is zero. Thus, whenever the preliminary process produces a negative outcome, excess zeros are produced. The `zip` and `zinb` commands fit such models.

27.9 Generalized linear models

The generalized linear model is

$$g\{E(y_j)\} = \mathbf{x}_j\boldsymbol{\beta}, \qquad y_j \sim F$$

where $g(\cdot)$ is called the link function and F is a member of the exponential family, both of which you specify before estimation. `glm` fits this model.

The GLM framework encompasses a surprising array of models known by other names, including linear regression, Poisson regression, exponential regression, and others. Stata provides dedicated estimation commands for many of these. For instance, Stata has `regress` for linear regression, `poisson` for Poisson regression, and `streg` for exponential regression, and that is not all the overlap.

`glm` by default uses maximum likelihood estimation and alternatively estimates via iterated reweighted least squares (IRLS) when the `irls` option is specified. For each family, F, there is a corresponding link function, $g(\cdot)$, called the canonical link, for which IRLS estimation produces results identical to maximum likelihood estimation. You can, however, match families and link functions as you wish, and when you match a family to a link function other than the canonical link, you obtain a different but valid estimator of the standard errors of the regression coefficients. The estimator you obtain is asymptotically equivalent to the maximum likelihood estimator, which, in small samples, produces slightly different results.

For example, the canonical link for the binomial family is logit. `glm, irls` with that combination produces results identical to the maximum-likelihood `logit` (and `logistic`) command. The binomial family with the probit link produces the probit model, but probit is not the canonical link here. Hence, `glm, irls` produces standard error estimates that differ slightly from those produced by Stata's maximum-likelihood `probit` command.

Many researchers feel that the maximum-likelihood standard errors are preferable to IRLS estimates (when they are not identical), but they would have a difficult time justifying that feeling. Maximum likelihood probit is an estimator with (solely) asymptotic properties; `glm, irls` with the binomial family and probit link is an estimator with (solely) asymptotic properties, and in finite samples, the standard errors differ a little.

Still, we recommend that you use Stata's dedicated estimators whenever possible. IRLS (the theory) and `glm, irls` (the command) are all encompassing in their generality, meaning that they rarely use the right jargon or provide things in the way you wish they would. The narrower commands, such as `logit`, `probit`, and `poisson`, focus on the issue at hand and are invariably more convenient.

`glm` is useful when you want to match a family to a link function that is not provided elsewhere.

`glm` also offers several estimators of the variance–covariance matrix that are consistent, even when the errors are heteroskedastic or autocorrelated. Another advantage of a `glm` version of a model over a model-specific version is that many of these VCE estimators are available only for the `glm` implementation. You can also obtain the ML–based estimates of the VCE from `glm`.

27.10 Choice models

Choice models are models for data with outcomes that are choices. For instance, we could model choices made by consumers who select a breakfast cereal from several different brands. Stata's choice model commands come in two varieties—-commands for modeling for discrete choices and commands for modeling rank-ordered alternatives. When each individual selects a single alternative, say, a shopper purchasing one box of cereal, the data are discrete choice data. When each individual ranks the choices, say, that shopper orders cereals from most favorite to least favorite, the data are rank-ordered data.

Commands for binary outcomes, categorical outcomes, panel data, multilevel models, Bayesian estimation, and more can be useful in modeling choice data in addition to other types of data; see [CM] **Intro 4**. The commands described below are designed specifically for choice data. Each of these commands allows alternative-specific covariates—covariates that differ across alternatives (cereals in our example) and possibly across cases (individuals). In addition, these models properly account for unbalanced data in which some individuals choose from only a subset of the alternatives.

27.10.1 Models for discrete choices

For discrete choice data, Stata provides conditional logit (McFadden's choice), multinomial probit, mixed logit, panel-data mixed logit, and nested logit regression. For an overview of these models, see [CM] **Intro 5**.

`cmclogit` fits McFadden's choice model, also known as conditional logistic regression. `cmclogit` relies on the independence of irrelevant alternatives (IIA) assumption, which implies that the relative probability of selecting alternatives should not change if we introduce or eliminate another alternative; see [CM] **Intro 8**.

The mixed logit model, the multinomial probit model, and the nested logit model relax the IIA assumption in different ways.

`cmmixlogit` fits a mixed logit regression for choice models. This model allows random coefficients on one or more of the alternative-specific predictors in the model. Through these random coefficients, the model allows correlation across alternatives and, thus, relaxes the IIA assumption. `cmxtmixlogit` extends this model for panel data.

`cmmprobit` fits a multinomial probit choice model. Like `cmclogit`, this command estimates fixed coefficients for all predictors. It does not require an IIA assumption because it directly models the correlation between the error terms for the different alternatives.

`nlogit` fits a nested logit choice model. With this model, similar alternatives—alternatives whose errors are likely to be correlated—can be grouped into nests. This model then accounts for correlation of alternatives within the same nest.

27.10.2 Models for rank-ordered alternatives

For rank-ordered alternatives, Stata provides the rank-ordered logit and rank-ordered probit model. For an overview of these models, see [CM] **Intro 6**.

`cmrologit` fits the rank-ordered logit model. This model is also known as the Plackett–Luce model, the exploded logit model, and choice-based conjoint analysis. This model requires the IIA assumption. It is unique because alternatives are not specified. They are instead identified only by the characteristics in alternative-specific covariates.

cmroprobit fits the rank-ordered probit model, an extension of the multinomial probit choice model for rank-ordered alternatives. It allows both alternative-specific and case-specific predictors. It does not assume IIA; instead, it models the correlation of errors across alternatives.

27.11 Exact estimators

Exact estimators refer to models that, rather than being estimated by asymptotic formulas, are estimated by enumerating the conditional distribution of the sufficient statistics and then computing the maximum likelihood estimate using that distribution. Standard errors cannot be estimated, but CIs can be and are obtained from the enumerations.

exlogistic fits logistic models of binary data in this way.

expoisson fits Poisson models of count data in this way.

In small samples, exact estimates have better coverage than the asymptotic estimates, and exact estimates are the only way to obtain estimates, tests, and CIs of covariates that perfectly predict the observed outcome.

27.12 Models with endogenous covariates

A covariate is endogenous if it is correlated with the unobservable components of a model. Endogeneity encompasses cases such as measurement error, omitted variables correlated with included regressors, and simultaneity. Stata offers several commands to address endogeneity depending on your outcome of interest and how you wish to model the correlation that generates the endogeneity problem.

Solutions to endogeneity rely on the use of instrumental variables. Instrumental variables are uncorrelated with the unobservable components of a model and are related to the outcome of interest only through their relationships with the endogenous variables.

Instrumental-variable models use instrumental variables to model endogeneity. Alternatively, a control function approach can be used. In this case, instrumental variables are used to directly model the correlation between unobservable components in the model.

ivregress fits linear outcome models with endogenous variables using the two-stage least-squares form of instrumental variables, the limited-information form of maximum likelihood, and a version of the generalized method of moments (GMM). The three estimators differ in the efficiency and robustness to additional assumptions such as constraints on the variances of the error terms.

ivprobit fits a probit outcome model where one or more of the covariates are endogenously determined. ivfprobit is like ivprobit but fits a model for a fractional outcome, such as a rate or proportion. ivtobit is like ivregress but allows for censored outcomes. ivpoisson fits a Poisson outcome model where one or more of the covariates are endogenously determined. It can also be used for modeling nonnegative continuous outcomes instead of counts. ivqregress fits a quantile regression model where one or more of the covariates are endogenously determined.

The GMM estimator implemented in ivregress is a special case of the estimators implemented in gmm. For other functional forms, you can write your own moment-evaluator program or supply the moment conditions as substitutable expressions to gmm; see **[U] 27.24 Generalized method of moments (GMM)**.

cfregress fits linear models with endogenous regressors using control functions. Endogenous variables are first modeled as a function of instruments using linear, probit, fractional probit, or Poisson regression. The residuals, or generalized residuals, from these first-stage regressions are then included in the main equation as control functions to make regression estimates robust to endogeneity. cfprobit is like cfregress but fits a probit outcome model for binary dependent variables.

The extended regression commands fit models with endogenous covariates that are binary, ordinal, or censored, as well as continuous. eregress fits a linear model with endogenous covariates, eintreg fits tobit and interval regression models with endogenous covariates, eprobit fits a probit model with endogenous covariates, and eoprobit fits an ordered probit model with endogenous covariates. You may also use these commands to accommodate endogenous sample selection (see **[U] 27.13 Models with endogenous sample selection**) and treatment effects (see **[U] 27.20 Causal inference**) in combination with endogenous covariates.

For systems of linear equations with endogenous covariates, the three-stage least-squares (3SLS) estimator in reg3 can produce constrained and unconstrained estimates. Structural equation models discussed in **[U] 27.25 Structural equation modeling (SEM)** and GMM estimators discussed in **[U] 27.24 Generalized method of moments (GMM)** are also widely used for such systems.

27.13 Models with endogenous sample selection

When unobservable factors that affect who is included in a sample are correlated with unobservable factors that affect the outcome, we say that there is endogenous sample selection. When present, endogenous sample selection should be modeled; consider using one of the commands discussed below.

What has become known as the Heckman model refers to linear regression in the presence of endogenous sample selection: $y_j = \mathbf{x}_j\boldsymbol{\beta}+\epsilon_j$ is not observed unless some event occurs that itself has probability $p_j = F(\mathbf{z}_j\boldsymbol{\gamma} + \nu_j)$, where ϵ and ν might be correlated and $\mathbf{z}_j$ and $\mathbf{x}_j$ may contain variables in common. heckman fits such models by maximum likelihood or Heckman's original two-step procedure.

This model has been generalized to replace the linear regression equation with another probit equation, and that model is fit by heckprobit. The command heckoprobit fits an ordered probit model in the presence of sample selection. Finally, heckpoisson is used to model count data subject to endogenous sample selection.

Stata's extended regression commands allow you to model endogenous sample selection along with endogenous covariates and treatment effects. These commands are discussed in **[U] 27.12 Models with endogenous covariates**.

27.14 Time-series models

ARIMA refers to models with autoregressive integrated moving-average processes, and Stata's arima command fits models with ARIMA disturbances via the Kalman filter and maximum likelihood. These models may be fit with or without covariates. arima also fits ARMA models.

ARFIMA, or autoregressive fractionally integrated moving average, handles long-memory processes. ARFIMA generalizes the ARMA and ARIMA models. ARMA models assume short memory; after a shock, the process reverts to its trend relatively quickly. ARIMA models assume shocks are permanent and memory never fades. ARFIMA provides a middle ground in the length of the process's memory. The arfima command fits ARFIMA models. In addition to one-step and dynamic forecasts, arfima can predict fractionally integrated series.

UCM, or unobserved components model, decomposes a time series into trend, seasonal, cyclic, and idiosyncratic components after controlling for optional exogenous variables. UCM provides a flexible and formal approach to smoothing and decomposition problems. The `ucm` command fits UCM models.

The estimated parameters of ARIMA, ARFIMA, and UCM are sometimes more easily interpreted in terms of the implied spectral density. `psdensity` transforms results.

Band-pass and high-pass filters are also used to decompose a time series into trend and cyclic components, even though the `tsfilter` commands are not estimation commands. Provided are Baxter–King, Butterworth, Christiano–Fitzgerald, and Hodrick–Prescott filters.

Stata's `prais` command performs regression with AR(1) disturbances using the Prais–Winsten or Cochrane–Orcutt transformation. Both two-step and iterative solutions are available, as well as a version of the Hildreth–Lu search procedure.

`newey` produces linear regression estimates with the Newey–West variance estimates that are robust to heteroskedasticity and autocorrelation of specified order.

Stata provides estimators for ARCH, GARCH, univariate, and multivariate models. These models are for time-varying volatility. ARCH models allow for conditional heteroskedasticity by including lagged variances. GARCH models also include lagged second moments of the innovations (errors). ARCH stands for "autoregressive conditional heteroskedasticity". GARCH stands for "generalized ARCH".

`arch` fits univariate ARCH and GARCH models, and the command provides many popular extensions, including multiplicative conditional heteroskedasticity. Errors may be normal or Student's t or may follow a generalized error distribution. Robust standard errors are optionally provided.

`mgarch` fits multivariate ARCH and GARCH models, including the diagonal vech model and the constant, dynamic, and varying conditional correlation models. Errors may be multivariate normal or multivariate Student's t. Robust standard errors are optionally provided.

Stata provides VAR, SVAR, instrumental-variables SVAR, and VEC estimators for modeling multivariate time series. VAR, SVAR, and instrumental-variables SVAR deal with stationary series. SVAR places additional constraints on the VAR model that identifies the impulse–response functions. Instrumental-variables SVAR uses instruments to identify the impulse–response functions. VEC is for cointegrating VAR models. VAR stands for "vector autoregression"; SVAR, for "structural VAR"; and VEC, for "vector error-correction" model.

`var` fits VAR models, `svar` fits SVAR models, `ivsvar` fits instrumental-variables SVAR models, and `vec` fits VEC models. These commands share many of the same features for specification testing, forecasting, and parameter interpretation; see [TS] **var intro** for `var`, `svar`, and `ivsvar`; [TS] **vec intro** for `vec`; and [TS] **irf** for all four impulse–response functions and forecast-error variance decomposition. For lag-order selection, residual analysis, and Granger causality tests, see [TS] **var intro** (for `var`, `svar`, and `ivsvar`) and [TS] **vec intro**.

`lpirf` and `ivlpirf` estimate impulse–response functions via local projections. Local projections provide a flexible alternative to the model-based impulse–response estimates obtained after fitting a VAR model. Local projections simplify estimation and hypothesis testing, and the corresponding CIs can have better small-sample properties than those of the VAR estimates. When the impulse variable of interest is an endogenous regressor, `ivlpirf` accounts for the endogeneity by using instrumental variables.

sspace estimates the parameters of multivariate state-space models using the Kalman filter. The state-space representation of time-series models is extremely flexible and can be used to estimate the parameters of many different models, including vector autoregressive moving-average (VARMA) models, dynamic-factor (DF) models, and structural time-series (STS) models. It can also solve some stochastic dynamic-programming problems.

dfactor estimates the parameters of dynamic-factor models. These flexible models for multivariate time series provide for a vector-autoregressive structure in both observed outcomes and unobserved factors. They also allow exogenous covariates for observed outcomes or unobserved factors.

Sometimes time-series data are characterized by shifts in the mean or variance. Linear autoregressive models may not adequately capture these peculiarities of the data. Stata provides Markov-switching and threshold models to fit such series.

Markov-switching models are used for series that transition over a finite set of unobserved states where the transitions occur according to a Markov process. The time of transition from one state to another and the duration between changes in state are random. By contrast, threshold models are used for series that transition over regions determined by threshold values. You can use the mswitch command to fit Markov-switching dynamic-regression (MSDR) and Markov-switching autoregression (MSAR) models. MSDR models can accommodate higher autoregressive lags than MSAR models because the state vector does not depend on the autoregressive lags in an MSDR model. You can use threshold to fit threshold regression models.

Bayesian estimation of the VAR models is available by using the bayes: var command. Bayesian estimation allows you to incorporate external information about model parameters often available in practice and provides more stable inference in the presence of many model parameters relative to the size of the data; see *Advantages of Bayesian VAR models* in [BAYES] **bayes: var**. After estimation using bayes: var, you can obtain Bayesian forecasts by using bayesfcast and perform IRF and FEVD analysis by using bayesirf.

27.15 Panel-data models

Commands in this class begin with the letters xt. You must xtset your data before you can use an xt command.

27.15.1 Continuous outcomes with panel data

xtreg fits models of the form

$$y_{it} = \mathbf{x}_{it}\boldsymbol{\beta} + \nu_i + \epsilon_{it}$$

xtreg can produce the between-regression estimator, the within-regression (fixed-effects) estimator, the generalized least-squares (GLS) random-effects (matrix-weighted average of between and within results) estimator, or the correlated random-effects estimator. It can also produce the maximum-likelihood random-effects estimator. wildbootstrap xtreg also produces the within-regression estimator but with wild cluster bootstrap p-values and CIs for robust inference.

xtgee fits population-averaged models, and it optionally provides robust estimates of variance. Moreover, xtgee allows other correlation structures. One of particular interest to those with a lot of data goes by the name "unstructured". The within-panel correlations are simply estimated in an unconstrained way. **[U] 27.15.4 Generalized linear models with panel data** will discuss this estimator further because it is not restricted to linear regression models.

xtfrontier fits stochastic production or cost frontier models for panel data. You may choose from a time-invariant model or a time-varying decay model. In both models, the nonnegative inefficiency term is assumed to have a truncated-normal distribution. In the time-invariant model, the inefficiency term is constant within panels. In the time-varying decay model, the inefficiency term is modeled as a truncated-normal random variable multiplied by a specific function of time. In both models, the idiosyncratic error term is assumed to have a normal distribution. The only panel-specific effect is the random inefficiency term.

xtheckman fits random-effects models that account for endogenous sample selection. Random effects are included in the equation for the main outcome and in the selection equation and are allowed to be correlated.

xtivreg contains the between-2SLS estimator, the within-2SLS estimator, the first-differenced-2SLS estimator, and two GLS random-effects-2SLS estimators to handle cases in which some of the covariates are endogenous.

xteregress fits random-effects models that account for any combination of endogenous covariates, endogenous sample selection, and nonrandom treatment assignment.

xtdidregress estimates the average treatment effect on the treated (ATET) from observational data by difference in differences (DID) or difference in difference in differences (DDD). The ATET of a binary or continuous treatment on a continuous outcome is estimated by fitting a linear model with time and panel fixed effects. xthdidregress is an extension of xtdidregress that, instead of one ATET, allows for multiple ATETs that vary over time and over treatment cohorts. Treatment cohorts are groups subject to treatment at different points in time.

xthtaylor uses instrumental-variables estimators to estimate the parameters of panel-data random-effects models of the form

$$y_{it} = \mathbf{X}_{1it}\boldsymbol{\beta}_1 + \mathbf{X}_{2it}\boldsymbol{\beta}_2 + \mathbf{Z}_{1i}\boldsymbol{\delta}_1 + \mathbf{Z}_{2i}\boldsymbol{\delta}_2 + u_i + e_{it}$$

The individual effects u_i are correlated with the explanatory variables $\mathbf{X}_{2it}$ and $\mathbf{Z}_{2i}$ but are uncorrelated with $\mathbf{X}_{1it}$ and $\mathbf{Z}_{1i}$, where $\mathbf{Z}_1$ and $\mathbf{Z}_2$ are constant within the panel.

xtgls produces GLS estimates for models of the form

$$y_{it} = \mathbf{x}_{it}\boldsymbol{\beta} + \epsilon_{it}$$

where you may specify the variance structure of ϵ_{it}. If you specify that ϵ_{it} is independent for all i's and t's, xtgls produces the same results as regress up to a small-sample degrees-of-freedom correction applied by regress but not by xtgls.

You may choose among three variance structures concerning i and three concerning t, producing a total of nine different models. Assumptions concerning i deal with heteroskedasticity and cross-sectional correlation. Assumptions concerning t deal with autocorrelation and, more specifically, AR(1) serial correlation.

In the jargon of GLS, the random-effects model fit by xtreg has exchangeable correlation within i—xtgls does not model this particular correlation structure. xtgee, however, does.

Alternative methods report the OLS coefficients and a version of the GLS variance–covariance estimator. xtpcse produces panel-corrected standard error (PCSE) estimates for linear cross-sectional time-series models, where the parameters are estimated by OLS or Prais–Winsten regression. When you are computing the standard errors and the variance–covariance estimates, the disturbances are, by default, assumed to be heteroskedastic and contemporaneously correlated across panels.

xtrc fits Swamy's random-coefficients linear regression model. In this model, rather than only the intercept varying across groups, all the coefficients are allowed to vary.

See **[U] 27.16 Multilevel mixed-effects models** for a generalization of xtreg and xtrc that allows for multiple levels of panels, random coefficients, and variance-component estimation in general. xtrc is a special case of mixed.

27.15.2 Censored outcomes with panel data

xttobit fits a random-effects tobit model and generalizes that to observation-specific censoring.

xtintreg performs random-effects interval regression and generalizes that to observation-specific censoring. Interval regression, in addition to allowing open-ended intervals, allows closed intervals.

xteintreg fits random-effects interval regression models that account for any combination of endogenous covariates, endogenous sample selection, and nonrandom treatment assignment.

These models are generalizable to multilevel data; see **[U] 27.16 Multilevel mixed-effects models**.

27.15.3 Discrete outcomes with panel data

xtprobit fits random-effects probit regression via maximum likelihood. It also fits population-averaged models via GEE.

xtlogit fits random-effects logistic regression models via maximum likelihood. It also fits conditional fixed-effects models via maximum likelihood and population-averaged models via GEE.

xtcloglog estimates random-effects complementary log–log regression via maximum likelihood. It also fits population-averaged models via GEE.

xteprobit fits random-effects probit models that account for any combination of endogenous covariates, endogenous sample selection, and nonrandom treatment assignment.

xtologit and xtoprobit are multiple-outcome models. xtologit fits a random-effects ordered logistic model, and xtoprobit fits a random-effects ordered probit model.

xteoprobit fits random-effects ordered probit models that account for any combination of endogenous covariates, endogenous sample selection, and nonrandom treatment assignment.

xtmlogit fits random-effects multinomial logistic regression models via maximum likelihood. It also fits conditional fixed-effects models via maximum likelihood.

xtpoisson fits two different random-effects Poisson regression models via maximum likelihood. The two distributions for the random effects are gamma and normal. xtpoisson also fits conditional fixed-effects models, and it fits population-averaged models via GEE.

xtnbreg fits random-effects negative binomial regression models via maximum likelihood (the distribution of the random effects is assumed to be beta). xtnbreg also fits conditional fixed-effects models, and it fits population-averaged models via GEE.

xtprobit, xtlogit, xtcloglog, xtpoisson, and xtnbreg are nothing more than xtgee with the appropriate family and link and an exchangeable error structure. See **[U] 27.15.4 Generalized linear models with panel data**.

These models are generalizable to multilevel data; see **[U] 27.16 Multilevel mixed-effects models**.

27.15.4 Generalized linear models with panel data

[U] 27.9 Generalized linear models discussed the model

$$g\{E(y_j)\} = \mathbf{x}_j\boldsymbol{\beta}, \qquad y_j \sim F$$

where $g(\cdot)$ is the link function and F is a member of the exponential family, both of which you specify before estimation.

There are two ways to extend the generalized linear model to panel data. They are the generalized linear mixed model (GLMM) and generalized estimation equations (GEE).

GEE uses a working correlation structure to model within-panel correlation. GEEs may be fit with the `xtgee` command.

For generalized linear models with multilevel data, including panel data, see [U] 27.16 Multilevel mixed-effects models.

27.15.5 Survival models with panel data

`xtstreg` fits a random-effects parametric survival-time model by maximum likelihood. The conditional distribution of the response given the random effects is assumed to be exponential, Weibull, lognormal, loglogistic, or gamma. Depending on the selected distribution, `xtstreg` can fit models using a proportional hazards (PH) or accelerated failure-time (AFT) parameterization. Unlike the other panel-data commands, `xtstreg` requires that the data be `xtset` and `stset`.

These models are generalizable to multilevel data; see [U] 27.16 Multilevel mixed-effects models.

27.15.6 Dynamic and autoregressive panel-data models

`xtregar` can produce the within estimator and a GLS random-effects estimator when the ϵ_{it} are assumed to follow an AR(1) process.

`xtabond` is for use with dynamic panel-data models (models in which there are lagged dependent variables) and can produce the one-step, one-step robust, and two-step Arellano–Bond estimators. `xtabond` can handle predetermined covariates, and it reports both the Sargan and autocorrelation tests derived by Arellano and Bond.

`xtdpdsys` is an extension of `xtabond` and produces estimates with smaller bias when the coefficients of the AR process are large. `xtdpdsys` is also more efficient than `xtabond`. Whereas `xtabond` uses moment conditions based on the differenced errors, `xtdpdsys` uses moment conditions based on both the differenced errors and their levels.

`xtdpd` is an extension of `xtdpdsys` and can be used to estimate the parameters of a broader class of dynamic panel-data models. `xtdpd` can be used to fit models with serially correlated idiosyncratic errors, whereas `xtdpdsys` and `xtabond` assume no serial correlation. `xtdpd` can also be used with models where the structure of the predetermined variables is more complicated than that assumed by `xtdpdsys` or `xtabond`.

`xtvar` is an extension of multivariate time-series regression for dynamic panel data. The parameters fit by `xtvar` are analogous to those of a VAR model. Therefore, although the estimation techniques are more similar to those of other dynamic panel-data models, the postestimation tools for testing, interpretation, and diagnostics are the same as those for multivariate time-series models. For instance, impulse–response functions (see [TS] **irf**) are useful for interpreting results.

27.15.7 Bayesian estimation

Bayesian estimation for some of the random-effects panel-data models is available via the `bayes` prefix command. See *Bayesian panel-data commands* in [BAYES] **Bayesian estimation** for the supported commands. You may be interested in Bayesian estimation, for instance, if you would like to incorporate external information about model parameters often available in practice or when you have small datasets and many parameters to estimate. Bayesian estimation is particularly useful if you are interested in estimating random effects because it provides an entire posterior distribution that incorporates all sources of variability in the model parameter estimates for each random effect. In addition, you can use the `bayespredict` command to compute predictions and their variability without relying on the assumption of asymptotic normality of the model parameter estimates. See *Panel-data models* in [BAYES] **bayes**.

27.16 Multilevel mixed-effects models

In multilevel data, observations—subjects, for want of a better word—can be divided into groups that have something in common. Perhaps the subjects are students, and the groups attended the same high school, or they are patients who were treated at the same hospital, or they are tractors that were manufactured at the same factory. Whatever they have in common, it may be reasonable to assume that the shared attribute affects the outcome being modeled.

With regard to students and high school, perhaps you are modeling later success in life. Some high schools are better (or worse) than others, so it would not be unreasonable to assume that the identity of the high school had an effect. With regard to patients and hospital, the argument is much the same if the outcome is subsequent health: some hospitals are better (or worse) than others, at least with respect to particular health problems. With regard to tractors and factory, it would hardly be surprising if tractors from some factories were more reliable than tractors from other factories.

Described above is two-level data. The first level is the student, patient, or tractor, and the second level is the high school, hospital, or factory. Observations are said to be nested within groups: students within a high school, patients within a hospital, or tractors within a factory.

Even though the effect on outcome is not directly observed, one can control for the effect if one is willing to assume that the effect is the same for all observations within a group and that, across groups, the effect is a random draw from a statistical distribution that is uncorrelated with the overall residual of the model and other group effects.

We have just described multilevel models.

A more complicated scenario might have three levels: students nested within teachers within a high school, patients nested within doctors within a hospital, or tractors nested within an assembly line within a factory.

An alternative to three-level hierarchical data is crossed data. We have workers and their occupation and the industry in which they work.

Stata provides a suite of multilevel estimation commands. The estimation commands are the following:

Command	Outcome variable	Equivalent to
mixed	continuous	linear regression
menl	continuous	nonlinear regression
metobit	censored	tobit regression
meintreg	censored	interval regression
meprobit	binary	probit regression
melogit	binary	logistic regression
mecloglog	binary	complementary log–log regression
meoprobit	ordered categorical	ordered probit regression
meologit	ordered categorical	ordered logistic regression
mepoisson	count	Poisson regression
menbreg	count	negative binomial regression
mestreg	survival-time	parametric survival-time regression
meglm	various	generalized linear models

The above estimators provide random intercepts and random coefficients and allow constraints to be placed on coefficients and on variance components. (menl does not allow constraints.)

See the *Stata Multilevel Mixed-Effects Reference Manual*; in particular, see [ME] **me**.

27.17 Survival analysis models

Commands are provided to fit Cox proportional hazards models, competing-risks regression, and several parametric survival models, including exponential, Weibull, Gompertz, lognormal, loglogistic, and generalized gamma. The command for Cox regression is stcox. Cox regressions may be fit to left- or right-censored survival-time data. For interval-censored survival-time data, stintcox may be used to fit semiparametric Cox proportional hazards models, and stmgintcox may be used to fit marginal Cox proportional hazards models to multiple-event data. Parametric models may be fit to right-censored survival-time data by using the streg command and to interval-censored survival-time data by using the stintreg command.

stcox and streg support single- or multiple-failure-per-subject data. The command for competing-risks regression, stcrreg, and stintreg support only single-failure data. Conventional, robust, bootstrap, and jackknife standard errors are available with all four commands, with the exception that for stcrreg, robust standard errors are the conventional standard errors.

Both the Cox model and the parametric models (as fit using Stata) allow for two additional generalizations. First, the models may be modified to allow for latent random effects, or frailties. Second, the models may be stratified in that the baseline hazard function may vary completely over a set of strata. The parametric models also allow for the modeling of ancillary parameters.

Competing-risks regression, as fit using Stata, is a useful alternative to Cox regression for datasets where more than one type of failure occurs, in other words, for data where failure events compete with one another. In such situations, competing-risks regression allows you to easily assess covariate effects on the incidence of the failure type of interest without having to make strong assumptions concerning the independence of failure types.

`stcox`, `stcrreg`, and `streg` require that the data be `stset` so that the proper response variables can be established. After you `stset` the data, the time/censoring response is taken as understood, and you need supply only the regressors (and other options) to `stcox`, `stcrreg`, and `streg`. With `stcrreg`, one required option deals with specifying which events compete with the failure event of interest that was previously `stset`. `stintcox`, `stintreg`, and `stmgintcox` require that you specify the interval-censored time variables with the command and thus ignore any `st` settings.

Stata also provides commands to estimate average treatment effects and average treatment effects on the treated from observational survival-time data. See **[U] 27.20 Causal inference**.

We discuss panel-data survival-time models in **[U] 27.15.5 Survival models with panel data**. These models generalize to multilevel data; see **[U] 27.16 Multilevel mixed-effects models**.

27.18 Meta-analysis

Meta-analysis is a statistical method for combining the results from several different studies that answer similar research questions. The goal of meta-analysis is to compare the study results and, if possible, provide a unified conclusion based on an overall estimate of the effect of interest. Stata provides a suite of commands for conducting meta-analysis.

Study-specific effect sizes and their corresponding standard errors are the two main components of meta-analysis. They are specified during the declaration step ([META] **meta data**) using `meta set` or `meta esize`; see [META] **meta set** and [META] **meta esize**.

Basic meta-analysis summary, which includes the overall effect-size estimate and its CI and heterogeneity statistics, can be displayed in a table ([META] **meta summarize**) or on a forest plot ([META] **meta forestplot**). Three meta-analysis models—random-effects, fixed-effects, and common-effect—and several estimation methods, such as restricted maximum likelihood and Mantel–Haenszel, are supported.

Heterogeneity or between-study variation arises frequently in meta-analysis. It can be explored graphically using subgroups in forest plots or using Galbraith or L'Abbé plots. See the `subgroup()` option in [META] **meta forestplot**, and see [META] **meta galbraithplot** and [META] **meta labbeplot**. It can be examined analytically via meta-regression and subgroup analysis. See [META] **meta regress** and the `subgroup()` option in [META] **meta summarize**. You can also use `meta summarize` or `meta forestplot` to perform cumulative meta-analysis by specifying the `cumulative()` option with the command. Leave-one-out meta-analysis can be done by specifying the `leaveoneout` option with these commands.

The presence of publication bias is another concern in meta-analysis. It typically arises when the decision of whether to publish the results of a study depends on the statistical significance of its results. Smaller studies with nonsignificant findings are commonly more prone to publication bias. Standard and contour-enhanced funnel plots ([META] **meta funnelplot**), tests for funnel-plot asymmetry ([META] **meta bias**), and the trim-and-fill method ([META] **meta trimfill**) can all be used to explore publication bias and assess its impact on meta-analysis results. More generally, `meta funnelplot` and `meta bias` are used to explore the so-called small-study effects, or the tendency of smaller studies to report different, often larger, effect sizes compared with larger studies.

When a study reports multiple effect sizes, you may use meta mvregress to perform multivariate meta-analysis while accounting for the dependence among the effect sizes. You may also incorporate covariates in your model and conduct a multivariate meta-regression to explore the multivariate heterogeneity of the effect sizes.

Additionally, if the dependence among effect sizes stems from a hierarchical or multilevel structure, you may use meta meregress to perform multilevel meta-analysis. With meta meregress, you can fit models with random intercepts and random slopes. If, however, you are interested in fitting only random-intercepts meta-analysis models, you can use meta multilevel, which provides a simpler syntax.

Also available in the meta suite are postestimation tools, such as bubble plots and various predictions. See [META] **meta regress postestimation**, [META] **meta mvregress postestimation**, and [META] **meta me postestimation**.

27.19 Spatial autoregressive models

Stata's Sp estimation commands fit spatial autoregressive (SAR) models, also known as simultaneous autoregressive models. The commands allow spatial lags of the dependent and independent variables and spatial autoregressive errors. In time-series analysis, lags refer to recent times. In spatial analysis, lags mean nearby areas.

An essential part of the model specification for SAR models is the formulation of spatial lags. Spatial lags are specified using spatial weighting matrices. Because of the potentially large dimensions of the weighting matrices, Stata provides commands for creating, using, and saving spatial weighting matrices.

Spatial models estimate indirect or spillover effects from one spatial unit (area) to another. The models estimate direct effects, too, just as nonspatial models would. Direct effects are the effects within a spatial unit. Viewing estimates of the direct effects, indirect effects, and total effects is done by running estat impact after any of the Sp estimation commands. estat impact makes interpreting results easy.

Datasets for SAR models contain observations on geographical areas or other units; the only requirement is some measure of distance that distinguishes which units are close to each other. Spatial data for geographic areas are typically based on shapefiles. The Sp system converts standard-format shapefiles to Stata .dta files so they can be merged with other Stata .dta datasets.

The Sp system will also work without shapefiles. Data can contain (x, y) coordinates, or data need not be geographic at all. For example, Sp can be used to analyze social networks.

Read [SP] **Intro** and the introductory sections that follow it for an overview of SAR models and a tutorial with examples for preparing your data and creating spatial weighting matrices.

The available Sp estimation commands are as follows:

Command	Description	Equivalent to
spregress, gs2sls	SAR with GS2SLS estimator	regress
spregress, ml	SAR with ML estimator	regress
spivregress	SAR with endogenous regressors	ivregress
spxtregress, fe	fixed-effects SAR for panel data	xtreg, fe
spxtregress, re	random-effects SAR for panel data	xtreg, re
spxtregress, re sarpanel	random-effects SAR alternative	

spregress, gs2sls and spivregress will fit multiple spatial lags of the dependent variable, multiple spatial autoregressive error terms, and multiple spatial lags of covariates. The other Sp estimation commands will fit only one spatial lag of the dependent variable and only one spatial autoregressive error term, but will allow multiple spatial lags of covariates.

27.20 Causal inference

Many research questions focus on causality. We wish to draw causal inferences when we ask what would happen to an outcome of interest when a change is made to another variable, often called a treatment variable. Under proper assumptions, many of Stata's estimation commands can be used for causal inference; see [CAUSAL] **Intro** for more information. Stata also offers estimation commands specifically designed for estimating treatment effects when causal inference is the research goal.

teffects, stteffects, eteffects, mediate, didregress, hdidregress, xtdidregress, xthdidregress, telasso, and cate estimate treatment effects from observational data.

A treatment effect is the change in an outcome caused by an individual getting one treatment instead of another. We can estimate average treatment effects, but not individual-level treatment effects, because we observe each individual getting only one or another treatment.

teffects, stteffects, eteffects, and telasso use methods that specify what the individual-level outcomes would be for each treatment level, even though only one of them can be realized. This approach is known as the potential-outcome framework. See [CAUSAL] **teffects intro** for a basic introduction to the key concepts associated with observational data analysis. See [CAUSAL] **teffects intro advanced** for a more advanced introduction that provides the intuition behind the potential-outcome framework. [CAUSAL] **stteffects intro** extends the concepts in the two earlier introductions to survival-time data.

mediate takes the potential-outcomes approach to causal mediation analysis. This estimator allows you to estimate the direct effect that a treatment has on the outcome as well as the indirect effect through a mediator variable.

didregress and xtdidregress can also be understood within the potential-outcome framework; however, these estimators differ from teffects, stteffects, eteffects in that they include group and time effects in the model. Including group and time effects controls for group and time unobservables that might bias effect estimates.

hdidregress and xthdidregress estimate treatment effects that may vary over time and over treatment cohorts. Treatment cohorts are groups subject to treatment at different points in time. Both of these commands are extensions of didregress and xtdidregress, but unlike the former they estimate multiple ATETs instead of only one. hdidregress and xthdidregress are commands for heterogeneous treatment-effect estimation.

Suppose we want to use observational data to learn about the effect of exercise on blood pressure. The potential-outcome framework provides the structure to estimate what would be the average effect of everyone exercising instead of everyone not exercising, an effect known as average treatment effect (ATE). Similarly, we can estimate the average effect, among those who exercise, of exercising instead of not exercising, which is known as the average treatment effect on the treated (ATET). Finally, we could estimate the average blood pressure that would be obtained if everyone exercised or if no one exercised, parameters known as potential-outcome means (POMs).

teffects can estimate the ATE, the ATET, and the POMs. The estimators implemented in teffects impose the structure of the potential-outcome framework on the data in different ways.

- Regression-adjustment estimators use models for the potential outcomes. See [CAUSAL] **teffects ra**.
- Inverse-probability-weighted estimators use models for treatment assignment. See [CAUSAL] **teffects ipw**.
- Augmented inverse-probability-weighted estimators and inverse-probability-weighted regression-adjustment estimators use models for the potential outcomes and for treatment assignment. These estimators have the double-robust property; they correctly estimate the treatment effect even if only one of the two models is correctly specified. See [CAUSAL] **teffects aipw** and [CAUSAL] **teffects ipwra**.
- Nearest-neighbor matching (NNM) and propensity-score matching (PSM) estimators compare the outcomes of individuals who are as similar as possible except that one gets the treatment and the other does not. NNM uses a nonparametric similarity measure, while PSM uses estimated treatment probabilities to measure similarity. See [CAUSAL] **teffects nnmatch** and [CAUSAL] **teffects psmatch**.

`stteffects` can estimate the ATE, the ATET, and the POMs. The estimators implemented in `stteffects` impose the structure of the potential-outcome framework on the data in different ways.

- Regression-adjustment estimators use models for the potential outcomes, and censoring is adjusted for the log-likelihood function. See [CAUSAL] **stteffects ra**.
- Inverse-probability-weighted estimators use models for treatment assignment and for the censoring time. See [CAUSAL] **stteffects ipw**.
- Inverse-probability-weighted regression-adjustment (IPWRA) estimators use models for the potential outcomes and for treatment assignment. IPWRA estimators can adjust for censoring in the outcome model or with a separate censoring model. These estimators have the double-robust property: they correctly estimate the treatment effect even if only the outcome model or the treatment-assignment model is correctly specified. If a censoring model is specified, both the treatment-assignment model and the censoring model must be correctly specified for the estimator to be double robust. See [CAUSAL] **stteffects ipwra**.
- Weighted regression-adjustment estimators model the outcome and the time to censoring. See [CAUSAL] **stteffects wra**.

`teffects` and `stteffects` can estimate treatment effects from multivalued treatments; see [CAUSAL] **teffects multivalued**.

`telasso` estimates the ATE, the ATET, and the POMs from observational data by augmented inverse-probability weighting while using lasso methods to select from potential control variables to be included in the model.

The `cate` suite of commands is useful for estimating the average treatment effects conditional on a set of variables, known as conditional average treatment effects (CATEs). `cate` provides three different CATE estimates: individualized average treatment effects (IATEs), group average treatment effects (GATEs), and sorted group average treatment effects (GATESs). IATEs are treatment effects conditional on observation-level characteristics. There is one IATE for each observation in the data. GATEs are treatment effects conditional on prespecified groups. There is a treatment effect for each group. GATESs compute average treatment effects for a prespecified number of groups. The groups are determined by quantiles of individual-level treatment-effects values. Estimating CATE allows us to study the treatment-effect heterogeneity and evaluate the treatment-assignment policy.

`mediate` decomposes the effect of the treatment on the outcome into direct effects, meaning how the treatment directly affects the outcome, and indirect effects, meaning how the treatment indirectly affects the outcome through a mediator. `mediate` estimates the average direct treatment effect (ADTE), average

indirect treatment effect (AITE), average direct treatment effect with respect to treatment (ADTET), average indirect treatment effect with respect to controls (AITEC), total average treatment effect (ATE), and the POMs.

didregress and xtdidregress estimate the ATET from observational data by difference in differences (DID) or difference in difference in differences (DDD). The ATET of a binary or continuous treatment on a continuous outcome is estimated by fitting a linear model with time and group fixed effects for didregress and time and panel fixed effects for xtdidregress. didregress is for repeated cross-sections in which different groups of individuals are observed at each time period. xtdidregress is for panel data. hdidregress and xthdidregress extend didregress and xtdidregress to estimate ATETs that may vary over time and over treatment cohorts. Treatment cohorts are groups subject to treatment at different points in time.

It is not appropriate to use teffects, stteffects, telasso, didregress, hdidregress, xtdidregress, or xthdidregress when a treatment is endogenously determined (the potential outcomes are not conditionally independent). When the treatment is endogenous, an endogenous treatment-effects model can be used to estimate the ATE. These models consider the effect of an endogenously determined binary treatment variable on the outcome.

eteffects can estimate the ATE, the ATET, and the POMs. It fits endogenous treatment-effects models by using either a linear or a nonlinear (probit, fractional probit, or exponential) model for the outcome. eteffects implements control-function regression-adjustment estimators.

etregress and etpoisson also fit endogenous treatment-effects models and can be used to estimate the ATE and the ATET. See [CAUSAL] **etregress** and [CAUSAL] **etpoisson**. etregress fits an endogenous treatment-effects model by using a linear model for the outcome. etpoisson fits an endogenous treatment-effects model by using a nonlinear (exponential) model for the outcome.

When the outcome is censored, eintreg estimates effects of endogenously or exogenously assigned treatments. eregress, eprobit, and eoprobit estimate effects of endogenously or exogenously assigned treatments, when the outcome is continuous, binary, or ordinal, respectively. All four commands can account for endogenous sample selection and endogenous covariates in combination with endogenous or exogenous treatment. See **[U] 27.13 Models with endogenous sample selection** and **[U] 27.12 Models with endogenous covariates**.

27.21 Pharmacokinetic data

There are four estimation commands designed for analyzing pharmacokinetic data. See [R] **pk** for an overview of the pk system.

1. pkexamine calculates pharmacokinetic measures from time-and-concentration subject-level data. pkexamine computes and displays the maximum measured concentration, the time at the maximum measured concentration, the time of the last measurement, the elimination time, the half-life, and the area under the concentration-time curve (AUC).
2. pksumm obtains the first four moments from the empirical distribution of each pharmacokinetic measurement and tests the null hypothesis that the distribution of that measurement is normally distributed.
3. pkcross analyzes data from a crossover design experiment. When one is analyzing pharmaceutical trial data, if the treatment, carryover, and sequence variables are known, the omnibus test for separability of the treatment and carryover effects is calculated.

4. pkequiv performs bioequivalence testing for two treatments. By default, pkequiv computes a CI for the ratio of treatment geometric means, calculated using log-scaled data. pkequiv can also calculate CIs using the original data, such as the classic shortest CI and a CI based on Fieller's theorem. Also, pkequiv performs interval hypothesis tests for bioequivalence, such as Schuirmann's two one-sided tests.

See [ME] **menl** for fitting pharmacokinetic models using nonlinear mixed-effects models.

27.22 Multivariate analysis

Stata's multivariate capabilities can be found in the *Multivariate Statistics Reference Manual*.

1. mvreg fits multivariate regressions.
2. manova provides MANOVA and MANCOVA (multivariate ANOVA and ANCOVA). The command fits MANOVA and MANCOVA models, one-way and up—including two-way factorial, three-way factorial, etc.—and it fits nested and mixed-design models.
3. canon estimates canonical correlations and their corresponding loadings. Canonical correlation attempts to describe the relationship between two sets of variables.
4. pca extracts principal components and reports eigenvalues and loadings. Some people consider principal components a descriptive tool—in which case standard errors as well as coefficients are relevant—and others look at it as a dimension-reduction technique.
5. factor fits factor models and provides principal factors, principal-component factors, iterated principal-component factors, and maximum likelihood solutions. Factor analysis is concerned with finding few common factors $\hat{\mathbf{z}}_k$, $k = 1, \ldots, q$, that linearly reconstruct the original variables $\mathbf{y}_i$, $i = 1, \ldots, L$.
6. tetrachoric, in conjunction with pca or factor, allows you to perform PCA or factor analysis on binary data.
7. rotate provides a wide variety of orthogonal and oblique rotations after factor and pca. Rotations are often used to produce more interpretable results.
8. procrustes performs Procrustes analysis, one of the standard methods of multidimensional scaling. It can perform orthogonal or oblique rotations as well as translation and dilation.
9. mds performs metric and nonmetric multidimensional scaling for dissimilarity between observations with respect to a set of variables. A wide variety of dissimilarity measures are available and, in fact, are the same as those for cluster.
10. ca performs correspondence analysis, an exploratory multivariate technique for analyzing cross-tabulations and the relationship between rows and columns.
11. mca performs multiple correspondence analysis (MCA) and joint correspondence analysis (JCA).
12. mvtest performs tests of multivariate normality along with tests of means, covariances, and correlations.
13. cluster provides cluster analysis; both hierarchical and partition clustering methods are available. Strictly speaking, cluster analysis does not fall into the category of statistical estimation. Rather, it is a set of techniques for exploratory data analysis. Stata's cluster environment has many different similarity and dissimilarity measures for continuous and binary data.

14. discrim and candisc perform discriminant analysis. candisc performs linear discriminant analysis (LDA). discrim also performs LDA, and it performs quadratic discriminant analysis (QDA), kth nearest neighbor (KNN), and logistic discriminant analysis. The two commands differ in default output. discrim shows the classification summary, candisc shows the canonical linear discriminant functions, and both will produce either.

For multivariate linear models that can include observed and latent variables, see **[U] 27.25 Structural equation modeling (SEM)**. To fit item response theory models to binary, ordinal, and nominal items, and their combinations, see **[U] 27.28 Item response theory (IRT)**. For multivariate time-series models, see **[U] 27.14 Time-series models**. To fit a multivariate meta-regression, see [META] **meta mvregress**.

27.23 Maximum likelihood estimation

Many of Stata's estimation commands fit models by using maximum likelihood estimation. If Stata does not have a command for the model you wish to fit and if you can specify the likelihood for the model, then you can use the mlexp command or the ml suite of commands to perform maximum likelihood estimation. mlexp is an easy-to-use command that does not require any programming; it does, however, require that you can write the log likelihood for each observation and that the overall log likelihood is the sum of the individual log likelihoods. ml can fit classes of models that do not meet these requirements, such as models for panel data; however, ml does require some programming.

27.24 Generalized method of moments (GMM)

gmm fits models using generalized method of moments (GMM). With the interactive version of the command, you enter your moment equations directly into the dialog box or command line using substitutable expressions just like with nl or nlsur. The moment-evaluator program version gives you greater flexibility in exchange for increased complexity; with this version, you write a program that calculates the moments based on a vector of parameters passed to it.

gmm can fit both single- and multiple-equation models, and you can combine moment conditions of the form $E\{\mathbf{z}_i u_i(\boldsymbol{\beta})\} = \mathbf{0}$, where $\mathbf{z}_i$ is a vector of instruments and $u_i(\boldsymbol{\beta})$ is often an additive regression error term, as well as more general moment conditions of the form $E\{\mathbf{h}_i(\mathbf{z}_i; \boldsymbol{\beta})\} = \mathbf{0}$. In the former case, you specify the expression for $u_i(\boldsymbol{\beta})$ and use the instruments() and xtinstruments() options to specify $\mathbf{z}_i$. In the latter case, you specify the expression for $\mathbf{h}_i(\mathbf{z}_i; \boldsymbol{\beta})$; because that expression incorporates your instruments, you do not use the instruments() or xtinstruments() option.

gmm supports cross-sectional, time-series, and panel data. You can request weight matrices and VCEs that are suitable for independent and identically distributed errors, that are suitable for heteroskedastic errors, that are appropriate for clustered observations, or that are heteroskedasticity- and autocorrelation-consistent (HAC). For HAC weight matrices and VCEs, gmm lets you specify the bandwidth or request an automatic bandwidth selection algorithm.

27.25 Structural equation modeling (SEM)

SEM stands for "structural equation modeling". The sem and gsem commands fit SEM.

sem fits standard linear SEMs. gsem fits what we call generalized SEMs, generalized to allow for generalized linear responses and multilevel modeling.

Generalized linear means, among other types of responses, binary responses such as probit and logit, count responses such as Poisson and negative binomial, categorical responses such as multinomial logit, ordered responses such as ordered probit and ordered logit, censored responses such as tobit, and survival responses such as exponential and Weibull. Generalized linear includes linear responses.

Multilevel modeling allows for nested effects, such as patient within doctor and patients within doctor within hospital, and crossed effects, such as occupation and industry.

Let's start with `sem`. `sem` can fit models ranging from linear regression to measurement models to simultaneous equations, including confirmatory factor analysis (CFA) models, correlated uniqueness models, latent growth models, and multiple indicators and multiple causes (MIMIC) models. You can obtain standardized or unstandardized results, direct and indirect effects, goodness-of-fit statistics, modification indices, score tests, Wald tests, linear and nonlinear tests of estimated parameters, and linear and nonlinear combinations of estimated parameters with CIs. You can perform estimation across groups with easy model specification and easy-to-use tests for group invariance. This can all be done using raw or summary statistics data. In addition, `sem` optionally can use full information maximum-likelihood (FIML) estimation to handle observations containing missing values.

`gsem` extends the types of models that can be fit. Responses may be continuous, ordinal, count, categorical, or survival time, and `gsem` allows for multilevel modeling. Latent variables can be included at any level. This allows for fitting models with random intercepts and random slopes. These random effects may be nested or crossed.

There is considerable overlap in the capabilities of `sem` and `gsem`. Whenever there is overlap, `sem` is faster and sometimes easier to use.

The generalized response variables allowed by `gsem` permit fitting measurement models with different types of responses, latent growth models with different types of responses, and so on.

`gsem` can also fit item response theory (IRT) models, multilevel CFA models, models for latent class analysis (LCA), finite mixture models (FMMs), multilevel mixed-effects models, and multilevel structural equation models. See **[U] 27.28 Item response theory (IRT)**, **[U] 27.26 Latent class models**, and **[U] 27.27 Finite mixture models (FMMs)**.

Where appropriate, results can be reported in exponentiated form to provide odds ratios, incidence-rate ratios, and relative-risk ratios. You can also obtain predictions, likelihood-ratio tests, Wald tests, predictive margins, contrasts, and pairwise comparisons.

Whether fitting a model with `sem` or `gsem`, you can specify your model by typing the command or by using the SEM Builder to draw path diagrams.

For those of you unfamiliar with SEM, it is worth your time to learn about it if you ever fit linear regressions, logistic regressions, ordered logit regressions, ordered probit regressions, Poisson regressions, seemingly unrelated regressions, multivariate regressions, simultaneous systems, measurement-error models, selection models, endogenous treatment-effects models, tobit models, survival models, fractional response models, or multilevel mixed-effects models.

You may also want to learn about SEM if you are interested in GMM. `sem` and `gsem` fit many of the same models by maximum likelihood and quasimaximum likelihood that you can fit by GMM.

sem and gsem can be used to fit many models that can be fit by other Stata commands. The advantage of using sem and gsem is in the extensions they can provide. They allow for introduction of latent variables to account for measurement error, simultaneous equations with different types of responses, multilevel versions of popular models such as selection models, and more.

See the *Stata Structural Equation Modeling Reference Manual*; in particular, see [SEM] **Intro 5**.

27.26 Latent class models

Latent class models (LCMs) are used to identify and understand unobserved groups in a population. Individuals in the population are assumed to be divided among these unobserved subpopulations called classes. The classes are represented by one or more categorical latent variables. LCMs often include a group of observed variables that are thought of as being measurements or indicators of class membership. The parameters in the models for these observed variables are allowed to vary across classes. In addition to modeling the observed variables, we also model the probability of being in each class.

After fitting an LCM, we can estimate the proportion of individuals in the population who belong to each class. We can also predict each individual's probability of belonging to each class.

We use LCM to refer to any model that includes categorical latent variables. In some literature, LCMs are more narrowly defined to include only categorical latent variables and the binary or categorical observed measurement variables, but we do not make such a restriction. Other labels closely associated with LCMs are latent class analysis, latent cluster models, latent cluster analysis, latent profile models, latent profile analysis, and finite mixture models. Each of these models can be fit as an LCM in Stata. See [SEM] **Intro 5**.

You fit latent class models in Stata by specifying the lclass() option with gsem. See the *Stata Structural Equation Modeling Reference Manual*; in particular, see [SEM] **Intro 1**, [SEM] **Intro 2**, [SEM] **Intro 5**, [SEM] **Example 50g**, and [SEM] **Example 52g**.

27.27 Finite mixture models (FMMs)

Finite mixture models (FMMs) are used to classify observations, to adjust for clustering, and to model unobserved heterogeneity. In finite mixture modeling, the observed data are assumed to belong to unobserved subpopulations called classes, and mixtures of probability densities or regression models are used to model the outcome of interest.

You can use FMMs to estimate the means and variances of the underlying densities for each unobserved subpopulation. Along with densities, they allow mixtures of regression models for continuous, binary, ordinal, categorical, count, fractional, and survival outcomes, where parameters are allowed to vary across subpopulations. You also can use FMMs to estimate each subpopulation's proportion in the overall population. In addition, FMMs allow the inclusion of covariates that model the probability of being in each subpopulation.

You fit FMMs in Stata by specifying the fmm prefix with the number of subpopulations; see [FMM] **fmm estimation** for models that can be specified as FMMs.

The *Stata Finite Mixture Models Reference Manual* provides complete documentation of Stata's finite mixture modeling features. See [FMM] **fmm intro** for an overview of FMMs and an introductory example.

27.28 Item response theory (IRT)

Item response theory (IRT) is used in the design, analysis, scoring, and comparison of tests and similar instruments whose purpose is to measure a latent trait. Latent traits cannot be measured directly because they are unobservable, but they can be quantified with an instrument. An instrument is simply a collection of items designed to measure a person's level of the latent trait. For example, a researcher interested in measuring mathematical ability (latent trait) may design a test (instrument) consisting of 100 questions (items).

When designing the instrument or analyzing data from the instrument, the researcher is interested in how each individual item relates to the trait and how the group of items as a whole relates to the trait. IRT models allow us to study these relationships.

Stata provides a suite of IRT estimation commands to fit a variety of models for binary responses and categorical responses. Models can also be combined. The available commands are the following:

Command	Description	Response
`irt 1pl`	One-parameter logistic model	binary
`irt 2pl`	Two-parameter logistic model	binary
`irt 3pl`	Three-parameter logistic model	binary
`irt grm`	Graded response model	categorical
`irt nrm`	Nominal response model	categorical
`irt pcm`	Partial credit model	categorical
`irt gpcm`	Generalized partial credit model	categorical
`irt rsm`	Rating scale model	categorical
`irt hybrid`	Hybrid IRT model	combination

A major concept in IRT is the item characteristic curve (ICC). The ICC maps the relationship between the latent trait and the probability that a person "succeeds" on a given item (individual test question). `irtgraph icc` can be used to plot the ICCs for items after any of the models above.

`irtgraph tcc` is used to plot the test characteristic curve (TCC), which shows the relationship between the expected score on the whole test and the latent trait. Plots of the item information and test information can be obtained with `irtgraph iif` and `irtgraph tif`.

Researchers are often interested in determining whether an instrument measures the latent trait in the same way for different groups. Multiple-group IRT models allow parameters to differ across groups and can be fit by adding the `group()` option to any of the `irt` commands.

See [IRT] **irt** for more information.

27.29 Dynamic stochastic general equilibrium (DSGE) models

DSGE models are time-series models used in economics for policy analysis and forecasting. The models are derived from macroeconomic theory and include multiple equations. A key feature of these models is that expectations of future variables affect variables today; this distinguishes DSGE models from other multivariate time-series models. Another key feature is that, being derived from theory, the parameters can usually be interpreted in terms of that theory.

The `dsge` and `dsgenl` commands fit DSGE models. `dsgenl` fits nonlinear DSGE models, and `dsge` fits linear DSGE models. See the *Stata Dynamic Stochastic General Equilibrium Models Reference Manual*; in particular, [DSGE] **Intro 1**.

Bayesian estimation of DSGE models is available by using the `bayes: dsge` and `bayes: dsgenl` commands (see [BAYES] **bayes: dsge**, [BAYES] **bayes: dsgenl**, and [DSGE] **Intro 9**). Bayesian estimation allows you to incorporate external information about model parameters often available in practice; see [DSGE] **Intro 9b**. After estimation using `bayes: dsge` or `bayes: dsgenl`, you can perform IRF analysis by using [BAYES] **bayesirf**.

27.30 Lasso

Lasso simultaneously performs model selection and estimation. The set of candidate models for which you may consider using lasso is much larger than what can be evaluated with traditional model selection techniques, such as comparisons of Akaike or Bayesian information criteria. Because it allows simultaneous model selection and estimation and is feasible for very large models, lasso is one of the most popular and widely used machine learning tools.

Lasso is a solution to a penalized optimization problem for continuous, binary, count, and survival-time outcomes. Without the penalty, lasso would give the same solutions as traditional likelihood-based estimators. The penalty forces some of the variables to be excluded from the model. In other words, the penalty is what determines the model selection properties of the lasso. For more information on the lasso penalty, see [LASSO] **lasso**.

Related to lasso are the elastic net and the square-root lasso estimators. Both the elastic net and the square-root lasso have the model selection and estimation characteristics of lasso. The difference between lasso, elastic net, and square-root lasso is how they penalize the model. The elastic net penalty yields an estimator that works better than lasso when groups of variables are highly correlated. The square-root lasso is equivalent to the lasso but allows for easier computation of the penalty parameters. For more information on elastic net and square-root lasso, see [LASSO] **elasticnet** and [LASSO] **sqrtlasso**.

With Stata, you may use `lasso`, `elasticnet`, and `sqrtlasso` to implement the estimators mentioned above and to do out-of-sample predictions. You may also use these commands with random subsamples of the data used for training, validation, and prediction. You can use `splitsample` to easily split your data into such subsamples.

You can also go beyond prediction. You can use lasso to obtain inferences with double-selection lasso, partialing-out lasso, and cross-fit partialing-out lasso. These estimators allow you to estimate effects and perform tests on coefficients for a fixed and known set of covariates, while also performing model selection using lasso for a potentially large set of control variables. The following inferential lasso commands fit models with continuous, binary, and count outcomes:

Command	Description
`dsregress`	Double-selection lasso linear regression
`dslogit`	Double-selection lasso logistic regression
`dspoisson`	Double-selection lasso Poisson regression
`poregress`	Partialing-out lasso linear regression
`pologit`	Partialing-out lasso logistic regression
`popoisson`	Partialing-out lasso Poisson regression
`poivregress`	Partialing-out lasso instrumental-variables regression
`xporegress`	Cross-fit partialing-out lasso linear regression
`xpologit`	Cross-fit partialing-out lasso logistic regression
`xpopoisson`	Cross-fit partialing-out lasso Poisson regression
`xpoivregress`	Cross-fit partialing-out lasso instrumental-variables regression
`telasso`	Treatment-effects estimation using lasso

27.31 Survey data

Stata's svy command fits statistical models for complex survey data. svy is a prefix command, so to obtain linear regression, you type

```
. svy: regress ...
```

or to obtain probit regression, you type

```
. svy: probit ...
```

but you must first type a svyset command to define the survey design characteristics. Prefix svy works with many estimation commands, and everything is documented together in the *Stata Survey Data Reference Manual*.

svy supports the following variance-estimation methods:

- Taylor-series linearization
- Bootstrap
- Balanced repeated replication (BRR)
- Jackknife
- Successive difference replication (SDR)

See [SVY] **Variance estimation** for details.

svy supports the following survey design characteristics:

- With- and without-replacement sampling
- Observation-level sampling weights
- Stage-level sampling weights
- Stratification
- Poststratification
- Clustering
- Multiple stages of clustering without replacement
- BRR and jackknife replication weights

See [SVY] **svyset** for details. For an application of the svy prefix with stage-level sampling weights, see example 6 in [ME] **meglm**.

Subpopulation estimation is available for all estimation commands.

Tabulations and summary statistics are also available, including means, proportions, ratios, and totals over multiple subpopulations and direct standardization of means, proportions, and ratios.

See [SVY] **Survey**.

27.32 Multiple imputation

Multiple imputation (MI) is a statistical technique for estimation in the presence of missing data. If you estimate the parameters of y on x_1, x_2, and x_3 using any of the other Stata estimation commands, parameters are estimated on the data for which y, x_1, x_2, and x_3 contain no missing values. This process is known as listwise or casewise deletion because observations for which any of y, x_1, x_2, or x_3 contain

missing values are ignored or, said differently, deleted from consideration. MI is a technique to recover the information in those ignored observations when the missing values are missing at random (MAR) or missing completely at random (MCAR). Data are MAR if the probability that a value is missing may depend on observed data but not on unobserved data. Data are MCAR if the probability of missingness is not even a function of the observed data.

MI is named for the imputations it produces to replace the missing values in the data. MI does not just form replacement values for the missing data; it produces multiple replacements. The purpose is not to create replacement values as close as possible to the true ones but to handle missing data in a way resulting in valid statistical inference.

There are three steps in an MI analysis. First, one forms M imputations for each missing value in the data. Second, one fits the model of interest separately on each of the M resulting datasets. Finally, one combines those M estimation results into the desired single result.

The `mi` command does this for you. It can be used with most of Stata's estimation commands, including with survey, survival, and panel and multilevel models. See [MI] **Intro**.

27.33 Power, precision, and sample-size analysis

Sample-size determination is important for planning a study. It helps allocate the necessary resources to achieve the research objective of a study.

When a study uses hypothesis testing to make inference about parameters of interest, power and sample-size (PSS) analysis is used to investigate the optimal allocation of study resources to increase the likelihood of detecting the desired magnitude of the effect of interest. Additionally, for studies in which the data gathering process will take place over an extended period of time, it is common to conduct interim analyses as data trickle in, rather than a single analysis once all the data have been collected. A popular framework for conducting interim analyses while preserving type I error is the group sequential design. Group sequential designs allow researchers to stop a trial early, at one of these interim analyses, if they find compelling evidence that a treatment is effective or ineffective.

When a study uses CIs for inference, precision and sample-size (PrSS) analysis is used to estimate the required sample size to achieve the desired precision of a CI in a future study.

27.33.1 Power and sample-size analysis

PSS analysis is used to plan studies that will use hypothesis testing for inference. For example, suppose that we want to design a study to evaluate a new drug for lowering blood pressure. We want to test whether the mean blood pressure of the experimental group, which will receive the new drug, is the same as the mean blood pressure of the control group, which will receive the old drug. The post hoc analysis will use a two-sample t test to test the difference between the two means. How many subjects do we need to enroll in our study to detect a difference between means that is of clinical importance? PSS analysis can answer this question.

PSS analysis can also answer other questions that may arise during the planning stage of a study. For example, what is the power of a test given an available sample size, and how likely is it to detect an effect of interest given limited study resources? The answers to these questions may help reduce the cost of a study by preventing an overpowered study or may help avoid wasting resources on an underpowered study.

See [PSS-2] **Intro (power)** for more information about PSS analysis.

The power command performs PSS analysis. It provides PSS analysis for comparison of means, variances, proportions, correlations, and contingency tables. It also provides PSS analysis for simple and multiple linear regression, logistic regression, and for survival analysis. One-sample, two-sample, and paired analyses of means, variances, proportions, and correlations are supported. Contingency table analyses may be performed for matched samples, $2 \times 2 \times K$ tables, or $2 \times J$ tables. For survival-time data, one-sample analysis is supported for Cox proportional hazards models; two-sample analysis is supported for parametric or nonparametric comparison of survivor functions.

The power command can also account for a cluster randomized design (CRD) for some analyses, such as one- and two-sample analyses of means and proportions. In a CRD, groups of subjects or clusters are randomized instead of individual subjects. As a result, observations within a cluster are usually correlated, which must be accounted for when performing PSS analysis.

You can also add your own PSS methods to the power command; see [PSS-2] ***power usermethod***.

power provides both tabular output and graphical output, or power curves; see [PSS-2] **power, table** and [PSS-2] **power, graph** for details.

See [PSS-2] **power** for a full list of supported methods and the description of the command.

You can work with power commands either interactively or via a convenient point-and-click interface; see [PSS-2] **GUI (power)** for details.

27.33.2 Precision and sample-size analysis

PrSS analysis is used to plan studies that will use CIs for inference. For example, suppose again that we want to design a study to evaluate a new drug for lowering blood pressure. We now want to estimate the difference in the mean blood pressure of the experimental group, which will receive the new drug, and the mean blood pressure of the control group, which will receive the old drug. We will compute a two-sided 95% CI for the difference between the two means. How many subjects do we need to enroll in our study to obtain a CI that is narrow enough to draw inferences that are meaningful? PrSS analysis can answer this question.

PrSS analysis can also answer other questions that may arise during the planning stage of a study. For example, what is the width of a CI that can be obtained given an available sample size, and how likely is it that we obtain a CI of a specific width given limited study resources? The answers to these questions may help reduce costs by limiting the number of subjects in a study. They may also help prevent completing a study only to find that it had too few subjects to obtain a CI narrow enough to be useful.

See [PSS-3] **Intro (ciwidth)** for more information about PrSS analysis.

The ciwidth command performs PrSS analysis. It provides PrSS analysis for CIs for a mean or a variance. It also provides PrSS analysis for the difference in two means from independent samples and the difference in two means from paired samples.

You can also add your own PrSS methods to the ciwidth command; see [PSS-3] ***ciwidth usermethod***.

ciwidth provides both tabular output and graphical output, or sample-size curves; see [PSS-3] **ciwidth, table** and [PSS-3] **ciwidth, graph** for details.

See [PSS-3] **ciwidth** for a full list of supported methods and the description of the command.

You can work with ciwidth commands either interactively or via a convenient point-and-click interface; see [PSS-3] **GUI (ciwidth)** for details.

27.33.3 Group sequential designs

Group sequential designs (GSDs) are a type of adaptive design that allow researchers to stop a trial early if they find compelling evidence that a treatment is effective or ineffective. Suppose that we want to design a study to test whether the chemotherapy medicine sunitinib is effective for treating tumors. Also, suppose that we anticipate data collection to take place over three years. Rather than performing a single analysis once all the data have been collected, we can perform interim analyses as data trickle in. Suppose we perform the interim analyses every six months; for each interim analysis, we can test whether we have enough evidence to claim that sunitinib is effective or ineffective. If we find strong evidence of efficacy at the first interim analysis, we can terminate the study and proceed to applying for regulatory approval sooner rather than later. On the other hand, if we find strong evidence of inefficacy, we can terminate the trial because of futility and avoid exposing additional participants to this inadequate treatment. GSDs provide criteria for making these decisions at each interim analysis.

GSDs allow us to test for efficacy and futility at each interim analysis. With the `gsbounds` command, we can compute stopping boundaries for GSDs, and with the `gsdesign` commands, we can also compute sample sizes for the interim analyses. Stopping boundaries and sample sizes can be obtained for a one-sample mean or proportion test, two-sample means or proportions test, and a log-rank test.

You can also add your own GSD methods to the `gsdesign` command; see [ADAPT] ***gsdesign user-method***.

The `gsbounds` command and `gsdesign` suite of commands provide both tabular output and graphical output. See [ADAPT] **gsdesign** for a full list of supported methods and the description of the suite of commands.

27.34 Bayesian analysis

Bayesian analysis is a statistical analysis that answers research questions about unknown parameters of statistical models by using probability statements. Bayesian analysis rests on the assumption that all model parameters are random quantities and are subject to prior knowledge. This assumption is in sharp contrast with more traditional, frequentist analysis where all parameters are considered unknown but fixed quantities.

Bayesian analysis is based on modeling and summarizing the posterior distribution of parameters conditional on the observed data. The posterior distribution is composed of a likelihood distribution of the data and the prior distribution of the model parameters. Many posterior distributions do not have a closed form and must be approximated using, for example, Markov chain Monte Carlo (MCMC) methods such as Metropolis–Hastings (MH) methods, the Gibbs method, or sometimes their combination. The convergence of MCMC must be verified before any inference can be made. Once convergence is established, model checking can be performed by comparing various aspects of the distribution of the observed data with those of data that are simulated based on the fitted Bayesian model.

In Bayesian analysis, marginal posterior distributions of parameters are used for inference. They are summarized using point estimators, such as posterior mean and median, and using interval estimators, such as equal-tailed credible intervals and highest-posterior density intervals.

Stata provides a suite of commands for conducting Bayesian analysis. Bayesian estimation ([BAYES] **Bayesian estimation**) consists of the `bayes` prefix for fitting a variety of Bayesian regression models and the `bayesmh` command for fitting general Bayesian models. Both commands offer three MCMC sampling methods: an adaptive MH sampling, a Gibbs sampling, or a combination of the two. You can choose from a variety of supported Bayesian models, including panel-data, mul-

tilevel, and time-series models, or you can program your own Bayesian models; see [BAYES] **bayes**, [BAYES] **bayesmh**, and [BAYES] **bayesmh evaluators**. You can also perform Bayesian variable selection; see [BAYES] **bayesselect**.

Convergence of MCMC can be assessed visually using `bayesgraph`, and Gelman–Rubin convergence diagnostics can be computed using `bayesstats grubin`. Model checking can be performed using `bayespredict` and `bayesstats ppvalues`. Marginal summaries can be obtained using `bayesstats summary`, and hypothesis testing can be performed using `bayestest`; see [BAYES] **Bayesian postestimation**. Special-interest postestimation commands are also available to produce Bayesian forecasts ([BAYES] **bayesfcast**) after VAR models ([BAYES] **bayes: var**), and IRF analysis is available after VAR and linear and nonlinear DSGE models ([BAYES] **bayes: dsge** and [BAYES] **bayes: dsgenl**.

See [BAYES] **Bayesian commands** for more information about commands and for a quick *Overview example*.

27.35 Bayesian model averaging

Instead of fitting a single model, you can use Bayesian model averaging (BMA) to combine the results from multiple plausible models according to Bayesian principles to account for model uncertainty. See *Brief motivation* in *Remarks and examples* in [BMA] **Intro**.

In the regression framework, model uncertainty amounts to the uncertainty of which predictors should be included in the regression model. The `bmaregress` command performs BMA for a linear regression and can be used for inference, prediction, or model selection; see [BMA] **bmaregress**.

After using `bmaregress`, you can check for convergence with `bmagraph pmp` and use it and `bmastats models` to examine which models are more likely given the assumed prior and observed data, that is, have higher posterior model probabilities; see [BMA] **bmagraph pmp** and [BMA] **bmastats models**. You can use the `bmastats pip` command to identify important predictors—predictors that have a high probability of being included in a model based on the prior information and observed data; see [BMA] **bmastats pip**. You can use the `bmagraph varmap` command to more conveniently visualize both the influential models and predictors; see [BMA] **bmagraph varmap**. And you can use `bmastats jointness` to explore the joint importance of predictors across the considered models—whether variables tend to appear together in the models or separately; see [BMA] **bmastats jointness**. `bmastats msize` and `bmagraph msize` can be used to explore the summaries and the distribution of the model size to explore the complexity of a BMA model; see [BMA] **bmastats msize** and [BMA] **bmagraph msize**.

The above features allow you to perform model-choice analysis, but you can also use BMA for prediction. The `bmapredict` command computes various posterior predictive summaries such as predictive means and credible intervals, simulates a new outcome, and generates outcome replications; see [BMA] **bmapredict**. You can then use the `bmastats lps` command to evaluate the predictive performance of your BMA model by using the log predictive-score; see [BMA] **bmastats lps**.

Finally, when parameter averaging is applicable, you can perform inference for the regression coefficients. As in standard Bayesian analysis, your coefficient estimate is not just a single value but an entire distribution. In BMA analysis, this distribution additionally accounts for model uncertainty. You can use the `bmagraph coefdensity` command to explore this distribution; see [BMA] **bmagraph coefdensity**. More generally, you can use the `bmacoefsample` command to simulate an MCMC sample of regression coefficients and other model parameters (see [BMA] **bmacoefsample**) and, after that, use many of the standard Bayesian tools for inference; see [BAYES] **bayesstats summary** and [BAYES] **bayesgraph** for details.

See the full list of postestimation features in [BMA] **BMA postestimation**.

See [BMA] **Intro** and [BMA] **BMA commands** for more information about BMA and Stata commands for BMA analysis.

27.36 H2O machine learning

Machine learning methods are often used to solve research and business problems focused on prediction when the problems require more sophisticated modeling approaches than, for instance, linear regression or generalized linear models. Ensemble decision tree methods, which combine multiple decision trees to improve predictive performance, are popular and effective machine learning methods for solving such problems. H2O is a scalable and distributed machine learning and predictive analytics platform that allows you to perform data analysis and machine learning, including ensemble decision tree methods such as random forest and gradient boosting machine.

The `h2oml` suite of Stata commands is a wrapper for H2O and provides end-to-end support for H2O machine learning analysis using ensemble decision tree methods. In addition to `h2oml`, the `_h2oframe` command provides several key subcommands that connect Stata to an H2O cluster, import a Stata dataset into an H2O frame, and provide various H2O data management; see [H2OML] **H2O setup**.

`h2oml` *gbm* and `h2oml` *rf* provide the suite of estimation commands that implement gradient boosting and random forest regression, binary classification, and multiclass classification. `h2oml gbregress` and `h2oml rfregress` perform respective gradient boosting and random forest regressions for continuous and count responses; `h2oml gbbinclass` and `h2oml rfbinclass` perform gradient boosting and random forest classifications for binary responses; and `h2oml gbmulticlass` and `h2oml rfmulticlass` perform gradient boosting and random forest classifications for categorical responses (with more than two categories).

Each of these estimation commands allows you to use a validation frame or perform cross-validation to control for overfitting. They allow you to tune a variety of hyperparameters and to stop early for better model performance. Many commands also offer specialized options. For instance, `h2oml gbregress` allows you to choose from loss functions including quantile, Huber, and Tweedie. See [H2OML] ***h2oml gbm*** and [H2OML] ***h2oml rf*** for details.

After estimation, the `h2omlest` suite of commands can be used to manage estimation results. For instance, `h2omlest store` can be used to store the current estimation results for later use.

A number of postestimation commands are available to obtain tuning and estimation summaries. For instance, `h2omlestat gridsummary` is useful to view the results after tuning and select an alternative model that is more parsimonious. And `h2omlgraph scorehistory` can be used to display various validation curves to help monitor overfitting.

For binary and multiclass classifications, several commands can be used to explore model performance such as the `h2omlestat confmatrix` command, which displays the confusion matrix. Additionally, `h2omlgraph prcurve` and `h2omlgraph roc` can be used to plot precision–recall and receiver operating characteristic (ROC) curves after binary classification, and `h2omlestat hitratio` can be used to produce a hit-ratio table after multiclass classification.

The ultimate goal of machine learning is to obtain accurate prediction of the response on the new data. To achieve this goal, we often evaluate the model predictive performance by using an external, testing dataset. The `h2omlpostestframe` command provides a convenient way to specify the desired testing frame to be used in all subsequent postestimation analyses.

Depending on the estimation method, regression or classification, the `h2omlpredict` command produces predictions of continuous and count responses or class probabilities and classes.

Machine learning methods are often treated as a black box, meaning that little attempt is made to understand the obtained predictions. To rectify this, `h2oml` provides several postestimation commands to help explain predictions. The `h2omlgraph varimp` command can be used to assess the overall importance of predictors in the model, whereas the `h2omlgraph shapvalues` and `h2omlgraph shapsummary` commands can be used to explore the impact of predictors on individual predictions.

Finally, the `h2omltree` command can be used to save a specific decision tree in a DOT file and plot it by using the open-source software Graphviz; see [H2OML] **DOT extension**.

For more details about postestimation commands, see [H2OML] **h2oml postestimation**.

For more information about the `h2oml` command and its use with real-world data, see [H2OML] **h2oml**.

27.37 Reference

Gould, W. W. 2011. Use poisson rather than regress; tell a friend. *The Stata Blog: Not Elsewhere Classified*. https://blog.stata.com/2011/08/22/use-poisson-rather-than-regress-tell-a-friend/.