Zelig Documentation

Release 5.0-1

The Zelig Team

August 27, 2014

Zelig is a framework for interfacing with a wide range of statistical models and analytic methods in a common and simple way. Above and beyond estimation, it adds considerable infrastructure to existing heterogeneous R implementations by translating coefficient estimates into interpretable quantities of interest and automating statistical procedures (e.g., bootstrapping) though an intelligible call structure.

To get started, we recommend following the *Installation and Quickstart* guide. More information about the software, including our technical vision and goals for the project, can be found at the *About Zelig* page.

To view the code-base, visit the source repository at https://github.com/IQSS/Zelig and for regular updates and release information be sure to follow us on twitter at @IQSS. We also recommend joining the Zelig Google Group, where we are encouraging users to ask questions, report bugs, and help others.

You can also find the PDF of the documentation here.

CHAPTER ONE

INSTALLATION AND QUICKSTART

This guide is designed to get you up and running with the current *beta* release of Zelig (5.0-1).

1.1 Installing R and Zelig

Before using Zelig, you will need to download and install both the R statistical program and the Zelig package:

Installing R

To install R, go to http://www.r-project.org/ Select the `CRAN` option from the left-hand menu (CRAN is the Comprehensive R Archive Network where all files related to R can be found). Pick a CRAN mirror closest to your current geographic location (there are multiple mirrors of this database in various locations, selecting the one closest to you will be sure to maximize your the speed of your download). Follow the instructions for downloading R for Linux, Mac OS X, or Windows.

Installing Zelig

Zelig 5 is not available on `CRAN` yet.

Beta Release

Beta releases are updated with the latest fixes and newest experimental features, and generally reflect a copy currently being tested before submission to CRAN. To download this release, enter the following into an R console:

```
install.packages("Zelig", type = "source", repos = "http://r.iq.harvard.edu/")
```

Development Release

Development versions contain the latest code in-development. This means that the development version contains the latest code which may not be fully tested. To download this release:

```
# This installs devtools package, if not already installed
install.packages("devtools")
# This loads devtools
library(devtools)
# This downloads Zelig 5.0-1 from the IQSS Github repo
install_github('IQSS/Zelig')
```

If you have successfully installed the program, you will see a the following message: *"DONE (Zelig5)"*.

1.2 Quickstart Guide

Now that we have successfully downloaded and installed Zelig from Github, we will load the package and walk through am example. The scenario is a simple one: imagine you want to estimate the distance a car needs to stop given it's speed and you have a dataset of spped and stopping distances of cars. Throughout the rest of this guide, we will walk you through building a statistical model from this data using Zelig.

Loading Zelig

First, we have to load Zelig into R. After installing both R and Zelig, open R and type:

```
library(Zelig)
```

Building Models

Now, lets build a statistical model that captures the relationship a cars stopping distance and speed, where distance is the outcome (dependent) variable and speed is the only explanatory (independent) variable. The first decision we must make is what statistical model to test for a relationship between a cars speed and distance required for it to come to a full stop. To do this, we plot the two variables in our dataset to visually inspect any potential relationship:

```
# Scatterplot of car speed and distance required for full stop
plot(cars$speed, cars$dist, main = "Scatterplot of Car Speed and Distance Required for Full Stop", yl
# Fit regression line to data
abline(lm(cars$dist ~ cars$speed), col = "firebrick")
```

Also included in the scatter plot is a "best-fit" regression line that indicates a positive and linear relationship between our two variables. This basic test coupled with the fact that our outcome variable (distance) is continuous our best choice for model to use is a least squares regression.

To fit this model to our data, we must first create Zelig least squares object, then specify our model, and finally regress distance on speed to estimate the relationship between speed and distance:

```
# load toy dataset (when you install R, example datasets are also installed)
data(cars)
# initialize Zelig5 least squares object
z5 <- zls$new()
# estimate ls model
z5$zelig(dist ~ speed, data = cars)
# you can now get model summary estimates
summary(z5)
```

```
## Model: 1
## Call:
## stats::lm(formula = dist ~ speed, data = .)
##
## Coefficients:
## (Intercept)        speed
##      -17.58         3.93
##
## Next step: Use 'setx' method
```

So what do our model estimates tell us? First off, we can see that the positive 3.93 estimate for speed suggests a positive relationship between speed and distance a car needs to stop. That is, the faster a car is going, the longer the distance it needs to come to a full stop. In particular, we would interpret this coefficient as a one unit increase in speed (e.g., mph) leads to a 3 unit increase in distance (e.g., miles) needed for a car to stop. This interpretation is not very intuitive, however, and we might be interested in answering a particular question such as how much more distance does a car need to stop if it traveling 30 versus 50 miles per hour.

Figure 1.1: Scatterplot

Zelig makes this simple, by automating the translation of model estimates in interpretable quantities of interest (more on this below) using Monte Carlo simulations. To get this process started we need to set explanatory variables in our model (i.e., speed) using the `$setx()` method:

```
# set speed to 30
z5$setx(speed = 30)

# set speed to 50
z5$setx1(speed = 50)
```

Now that we've set our variables, all we have to do is run our simulations:

```
# run simulations and estimate quantities of interest
z5$sim()
z5

##
##  sim x :
##  -----
## ev
##    mean    sd   50%  2.5% 97.5%
## 1 100.4 6.263 100.3 88.07 112.9
## pv
##    mean    sd   50%  2.5% 97.5%
## 1 100.4 6.263 100.3 88.07 112.9
##
##  sim x1 :
##  -----
## ev
##    mean    sd   50%  2.5% 97.5%
## 1 179.1 13.98 178.8 150.9 206.9
## pv
##    mean    sd   50%  2.5% 97.5%
## 1 179.1 13.98 178.8 150.9 206.9
## fd
##    mean    sd   50%  2.5% 97.5%
## 1 78.66 7.942 78.58 62.82 94.18
```

Now we've estimated a model and calculated interpretable estimates at two speeds (30 versus 50 mph). What can we do with them? Zelig gives you access to estimated quantities of interest and makes plotting and presenting them particularly easy.

Quantities of Interest

As mentioned earlier, a major feature of Zelig is the translation of model estimates into easy to interpret quantities of interest (QIs). These QIs (e.g., expected and predicted values) can be accessed via the `$sim.out` field:

```
z5$sim.out

## $x
## Source: local data frame [1 x 2]
## Groups: <by row>
##
##               ev            pv
## 1 <dbl[1000,1]> <dbl[1000,1]>
##
## $x1
## Source: local data frame [1 x 3]
```

```
## Groups: <by row>
##
##              ev             pv             fd
## 1 <dbl[1000,1]> <dbl[1000,1]> <dbl[1000,1]>
```

Plots

A second major Zelig feature is how easy it is to plot QIs for presentation in slides or an article. Using the `plot()` function on the `z5$s.out` will produce ready-to-use plots with labels and confidence intervals.

Plots of QI's:

```
z5$graph()
```

Help

Finally, model documentation can be accessed using the `z5$help()` method after a model object has been initialized:

```
# documentation for least squares model
z5 <- zls$new()
z5$help()

# documentation for logitstic regression
z5 <- zlogit$new()
z5$help()
```

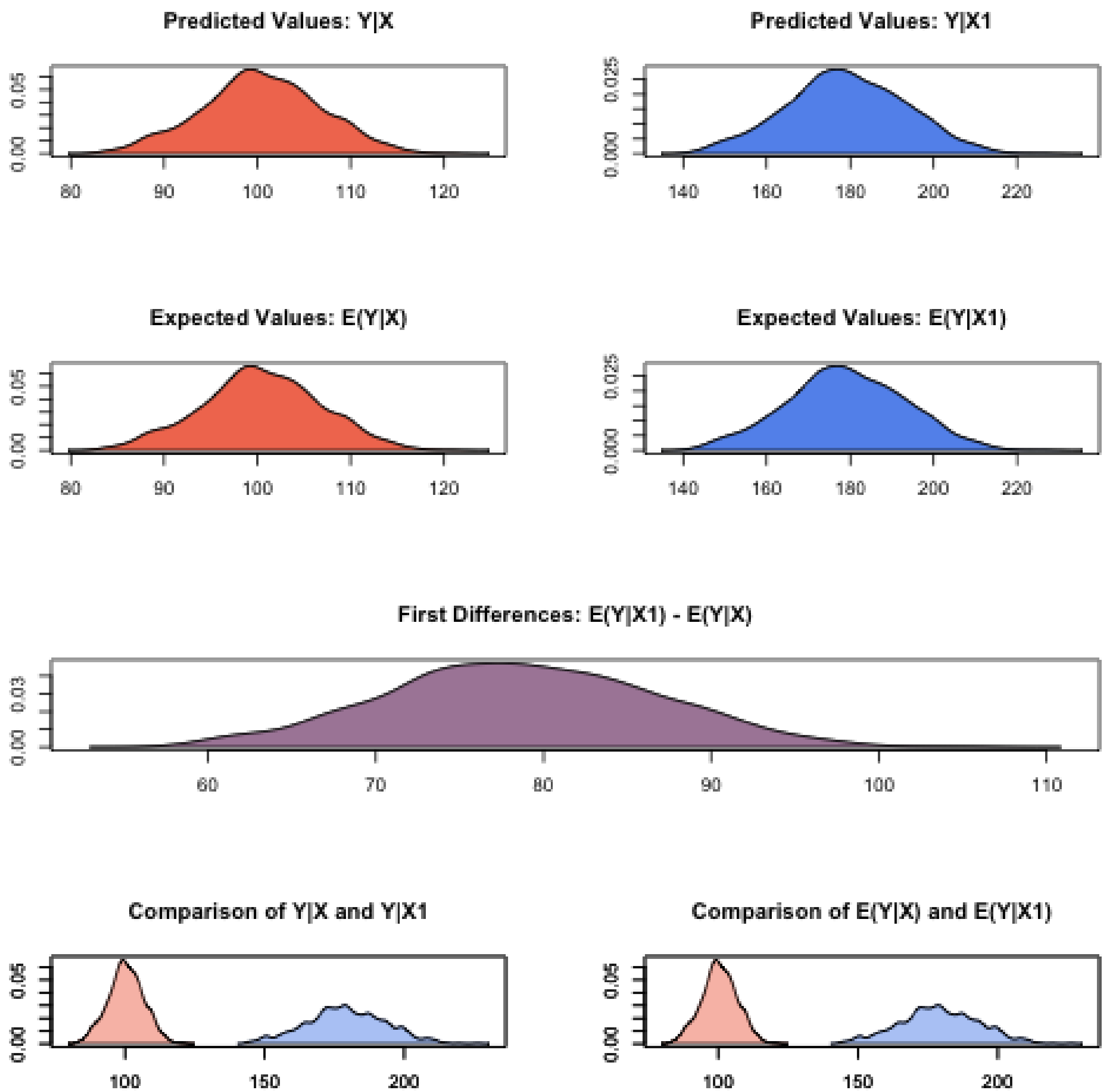


Figure 1.2: QIs

CHAPTER

TWO

MODEL REFERENCE AND VIGNETTES

This section includes technical information on the models currently implemented in Zelig (5.0-1). This includes a reference with a list of supported models as well as individual model vignettes with detailed information on the model, quantities of interest and syntax.

2.1 Reference

The following models are currently supported in Zelig 5.0-1:

- *Exponential Regression*: `zexp$new()`
- *Gamma Regression*: `zgamma()`
- *Logistic Regression*: `zlogit$new()`
- *Log Normal Regression*: `zlognorm$new()`
- *Least Squares Regression*: `zls$new()`
- *Negative Binomial Regression*: `zbinom$new()`
- *Normal Regression*: `znormal$new()`
- *Poisson Regression*: `zpoisson$new()`
- *Probit Regression*: `zprobit$new()`
- *Rare Events Logistic Regression*: `zrelogit$new()`
- *Tobit Regression*: `ztobit$new()`

2.2 zelig-exp

Exponential Regression for Duration Dependent Variables

Use the exponential duration regression model if you have a dependent variable representing a duration (time until an event). The model assumes a constant hazard rate for all events. The dependent variable may be censored (for observations have not yet been completed when data were collected).

2.2.1 Syntax

With reference classes:

```
z5 <- zexp$new()
z5$zelig(Surv(Y, C) ~ X, data = mydata)
z5$setx()
z5$sim()
```

With the Zelig 4 compatibility wrappers:

```
z.out <- zelig(Surv(Y, C) ~ X, model = "exp", data = mydata)
x.out <- setx(z.out)
s.out <- sim(z.out, x = x.out)
```

Exponential models require that the dependent variable be in the form Surv(Y, C), where Y and C are vectors of length n. For each observation i in 1, . . . , n, the value y_i is the duration (lifetime, for example), and the associated c_i is a binary variable such that $c_i = 1$ if the duration is not censored (*e.g.*, the subject dies during the study) or $c_i = 0$ if the duration is censored (*e.g.*, the subject is still alive at the end of the study and is know to live at least as long as y_i). If c_i is omitted, all Y are assumed to be completed; that is, time defaults to 1 for all observations.

2.2.2 Input Values

In addition to the standard inputs, zelig() takes the following additional options for exponential regression:

- robust: defaults to FALSE. If TRUE, zelig() computes robust standard errors based on sandwich estimators (see and) and the options selected in cluster.
- cluster: if robust = TRUE, you may select a variable to define groups of correlated observations. Let x3 be a variable that consists of either discrete numeric values, character strings, or factors that define strata. Then

```
z.out <- zelig(y ~ x1 + x2, robust = TRUE, cluster = "x3",
               model = "exp", data = mydata)
```

means that the observations can be correlated within the strata defined by the variable x3, and that robust standard errors should be calculated according to those clusters. If robust = TRUE but cluster is not specified, zelig() assumes that each observation falls into its own cluster.

2.2.3 Example

Attach the sample data:

```
data(coalition)
```

Estimate the model:

```
z.out <- zelig(Surv(duration, ciep12) ~ fract + numst2, model = "exp", data = coalition)

## How to cite this model in Zelig:
##   Olivia Lau, Kosuke Imai, Gary King. 2011.
##   exp: Exponential Regression for Duration Dependent Variables
##   in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
##   http://datascience.iq.harvard.edu/zelig
```

View the regression output:

```
summary(z.out)

## Model: 1Call:
## survival::survreg(formula = Surv(duration, ciep12) ~ fract +
##     numst2, data = ., dist = "exponential", model = FALSE)
##
## Coefficients:
## (Intercept)       fract      numst2
##    5.535873   -0.003909    0.461179
##
## Scale fixed at 1
##
## Loglik(model)= -1077   Loglik(intercept only)= -1101
##  Chisq= 46.66 on 2 degrees of freedom, p= 7.4e-11
## n= 314
## Next step: Use 'setx' method
```

Set the baseline values (with the ruling coalition in the minority) and the alternative values (with the ruling coalition in the majority) for X:

```
x.low <- setx(z.out, numst2 = 0)
x.high <- setx(z.out, numst2 = 1)
```

Simulate expected values and first differences:

```
s.out <- sim(z.out, x = x.low, x1 = x.high)
```

Summarize quantities of interest and produce some plots:

```
summary(s.out)

##
##  sim x :
##  -----
## ev
##    mean    sd   50%  2.5% 97.5%
## 1 15.35 1.529 15.29 12.71 18.36
## pv
##       mean   sd   50%   2.5% 97.5%
## [1,] 15.49 15.8 10.57 0.3662 53.73
##
##  sim x1 :
##  -----
## ev
##    mean    sd  50%  2.5% 97.5%
## 1 24.35 2.029 24.3 20.65 28.56
## pv
##       mean    sd   50%   2.5% 97.5%
## [1,] 25.65 25.85 16.46 0.4663 95.32
## fd
##    mean    sd  50%  2.5% 97.5%
## 1 9.001 2.515 9.06 3.816 13.91
```

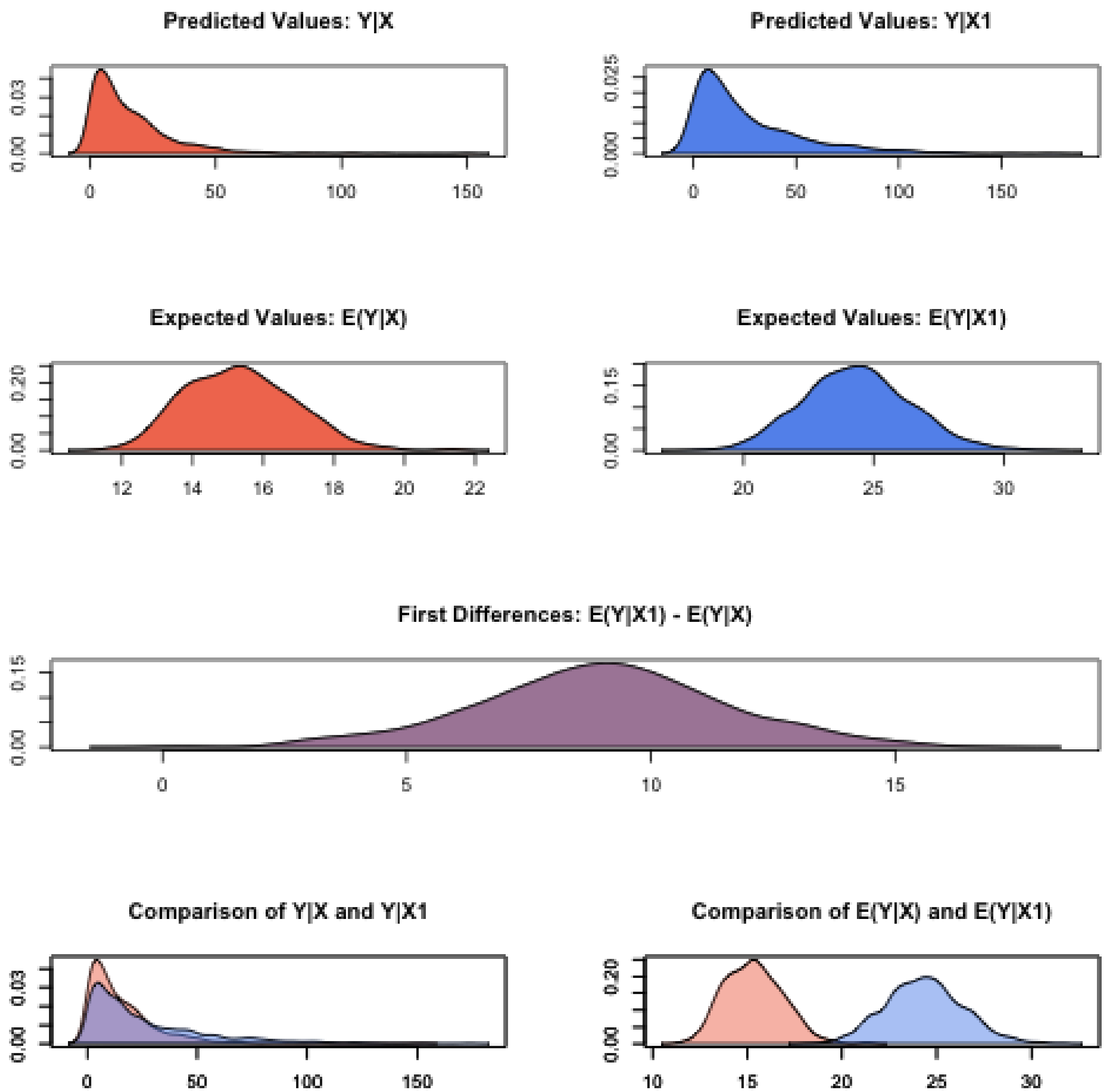


Figure 2.1: Zelig-exp

```
plot(s.out)
```

2.2.4 Model

Let Y_i^* be the survival time for observation i. This variable might be censored for some observations at a fixed time y_c such that the fully observed dependent variable, Y_i, is defined as

$$Y_i = \begin{cases} Y_i^* & \text{if } Y_i^* \leq y_c \\ y_c & \text{if } Y_i^* > y_c \end{cases}$$

- The *stochastic component* is described by the distribution of the partially observed variable Y^*. We assume Y_i^* follows the exponential distribution whose density function is given by

  $$f(y_i^* \mid \lambda_i) = \frac{1}{\lambda_i} \exp\left(-\frac{y_i^*}{\lambda_i}\right)$$

  for $y_i^* \geq 0$ and $\lambda_i > 0$. The mean of this distribution is λ_i.

  In addition, survival models like the exponential have three additional properties. The hazard function $h(t)$ measures the probability of not surviving past time t given survival up to t. In general, the hazard function is equal to $f(t)/S(t)$ where the survival function $S(t) = 1 - \int_0^t f(s)ds$ represents the fraction still surviving at time t. The cumulative hazard function $H(t)$ describes the probability of dying before time t. In general, $H(t) = \int_0^t h(s)ds = -\log S(t)$. In the case of the exponential model,

  $$\begin{aligned} h(t) &= \frac{1}{\lambda_i} \\ S(t) &= \exp\left(-\frac{t}{\lambda_i}\right) \\ H(t) &= \frac{t}{\lambda_i} \end{aligned}$$

  For the exponential model, the hazard function $h(t)$ is constant over time. The Weibull model and lognormal models allow the hazard function to vary as a function of elapsed time (see and respectively).

- The *systematic component* λ_i is modeled as

  $$\lambda_i = \exp(x_i \beta),$$

  where x_i is the vector of explanatory variables, and β is the vector of coefficients.

2.2.5 Quantities of Interest

- The expected values (qi$ev) for the exponential model are simulations of the expected duration given x_i and draws of β from its posterior,

  $$E(Y) = \lambda_i = \exp(x_i \beta).$$

- The predicted values (qi$pr) are draws from the exponential distribution with rate equal to the expected value.
- The first difference (or difference in expected values, qi$ev.diff), is

  $$\text{FD} = E(Y \mid x_1) - E(Y \mid x),$$

  where x and x_1 are different vectors of values for the explanatory variables.

- In conditional prediction models, the average expected treatment effect (att.ev) for the treatment group is

$$\frac{1}{\sum_{i=1}^{n} t_i} \sum_{i:t_i=1}^{n} \left\{ Y_i(t_i = 1) - E[Y_i(t_i = 0)] \right\},$$

  where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. When $Y_i(t_i = 1)$ is censored rather than observed, we replace it with a simulation from the model given available knowledge of the censoring process. Variation in the simulations is due to two factors: uncertainty in the imputation process for censored y_i^* and uncertainty in simulating $E[Y_i(t_i = 0)]$, the counterfactual expected value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.

- In conditional prediction models, the average predicted treatment effect (att.pr) for the treatment group is

$$\frac{1}{\sum_{i=1}^{n} t_i} \sum_{i:t_i=1}^{n} \left\{ Y_i(t_i = 1) - Y_i(\widehat{t_i = 0}) \right\},$$

  where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. When $Y_i(t_i = 1)$ is censored rather than observed, we replace it with a simulation from the model given available knowledge of the censoring process. Variation in the simulations is due to two factors: uncertainty in the imputation process for censored y_i^* and uncertainty in simulating $Y_i(\widehat{t_i = 0})$, the counterfactual predicted value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.

2.2.6 Output Values

The output of each Zelig command contains useful information which you may view. For example, if you run `z.out <- zelig(Surv(Y, C) ~ X, model = exp, data)`, then you may examine the available information in `z.out` by using `names(z.out)`, see the coefficients by using z.out$coefficients, and a default summary of information through `summary(z.out)`.

2.2.7 See also

The exponential function is part of the survival library by Terry Therneau, ported to R by Thomas Lumley. Advanced users may wish to refer to `help(survfit)` in the survival library.

2.3 zelig-gamma

Gamma Regression for Continuous, Positive Dependent Variables

Use the gamma regression model if you have a positive-valued dependent variable such as the number of years a parliamentary cabinet endures, or the seconds you can stay airborne while jumping. The gamma distribution assumes that all waiting times are complete by the end of the study (censoring is not allowed).

2.3.1 Syntax

With reference classes:

```
z5 <- zgamma$new()
z5$zelig(Y ~ X1 + X ~ X, data = mydata)
z5$setx()
z5$sim()
```

With the Zelig 4 compatibility wrappers:

```
z.out <- zelig(Y ~ X1 + X2, model = "gamma", data = mydata)
x.out <- setx(z.out)
s.out <- sim(z.out, x = x.out, x1 = NULL)
```

2.3.2 Example

Attach the sample data:

```
data(coalition)
```

Estimate the model:

```
z.out <- zelig(duration ~ fract + numst2, model = "gamma", data = coalition)

## How to cite this model in Zelig:
##   Kosuke Imai, Gary King, Olivia Lau. 2007.
##   gamma: Gamma Regression for Continuous, Positive Dependent Variables
##   in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone’s Statistical Software,"
##   http://datascience.iq.harvard.edu/zelig
```

View the regression output:

```
summary(z.out)

## Model: 1
## Call:  stats::glm(formula = duration ~ fract + numst2, family = Gamma("inverse"),
##     data = .)
##
## Coefficients:
## (Intercept)        fract       numst2
##   -0.012960     0.000115    -0.017387
##
## Degrees of Freedom: 313 Total (i.e. Null);  311 Residual
## Null Deviance:       301
## Residual Deviance: 272   AIC: 2430
## Next step: Use ’setx’ method
```

Set the baseline values (with the ruling coalition in the minority) and the alternative values (with the ruling coalition in the majority) for X:

```
x.low <- setx(z.out, numst2 = 0)
x.high <- setx(z.out, numst2 = 1)
```

Simulate expected values (qi$ev) and first differences (qi$fd):

```
s.out <- sim(z.out, x = x.low, x1 = x.high)

summary(s.out)

##
##  sim x :
```

```
##  -----
## ev
##       mean    sd   50%  2.5% 97.5%
## [1,] 14.43 1.122 14.33 12.51 16.97
## pv
##       mean    sd   50%  2.5% 97.5%
## [1,] 14.01 12.42 10.22 0.613  46.1
##
##  sim x1 :
##  -----
## ev
##       mean    sd   50%  2.5% 97.5%
## [1,] 19.22 1.094 19.18 17.22 21.58
## pv
##       mean    sd   50%   2.5% 97.5%
## [1,] 19.21 17.47 14.08 0.7479 64.43
## fd
##       mean   sd   50%  2.5% 97.5%
## [1,] 4.789 1.55 4.821 1.772 7.908
```

```
plot(s.out)
```

2.3.3 Model

- The Gamma distribution with scale parameter α has a *stochastic component*:

$$\begin{aligned} Y \sim & \quad \text{Gamma}(y_i \mid \lambda_i, \alpha) \\ f(y) = & \quad \frac{1}{\alpha^{\lambda_i} \, \Gamma \lambda_i} \, y_i^{\lambda_i - 1} \exp - \left\{ \frac{y_i}{\alpha} \right\} \end{aligned}$$

  for $\alpha, \lambda_i, y_i > 0$.

- The *systematic component* is given by

$$\lambda_i = \frac{1}{x_i \beta}$$

2.3.4 Quantities of Interest

- The expected values (qi$ev) are simulations of the mean of the stochastic component given draws of α and β from their posteriors:

$$E(Y) = \alpha \lambda_i.$$

- The predicted values (qi$pr) are draws from the gamma distribution for each given set of parameters (α, λ_i).
- If x1 is specified, sim() also returns the differences in the expected values (qi$fd),

$$E(Y \mid x_1) - E(Y \mid x)$$

  .

- In conditional prediction models, the average expected treatment effect (att.ev) for the treatment group is

$$\frac{1}{\sum_{i=1}^n t_i} \sum_{i:t_i=1}^n \left\{ Y_i(t_i = 1) - E[Y_i(t_i = 0)] \right\},$$

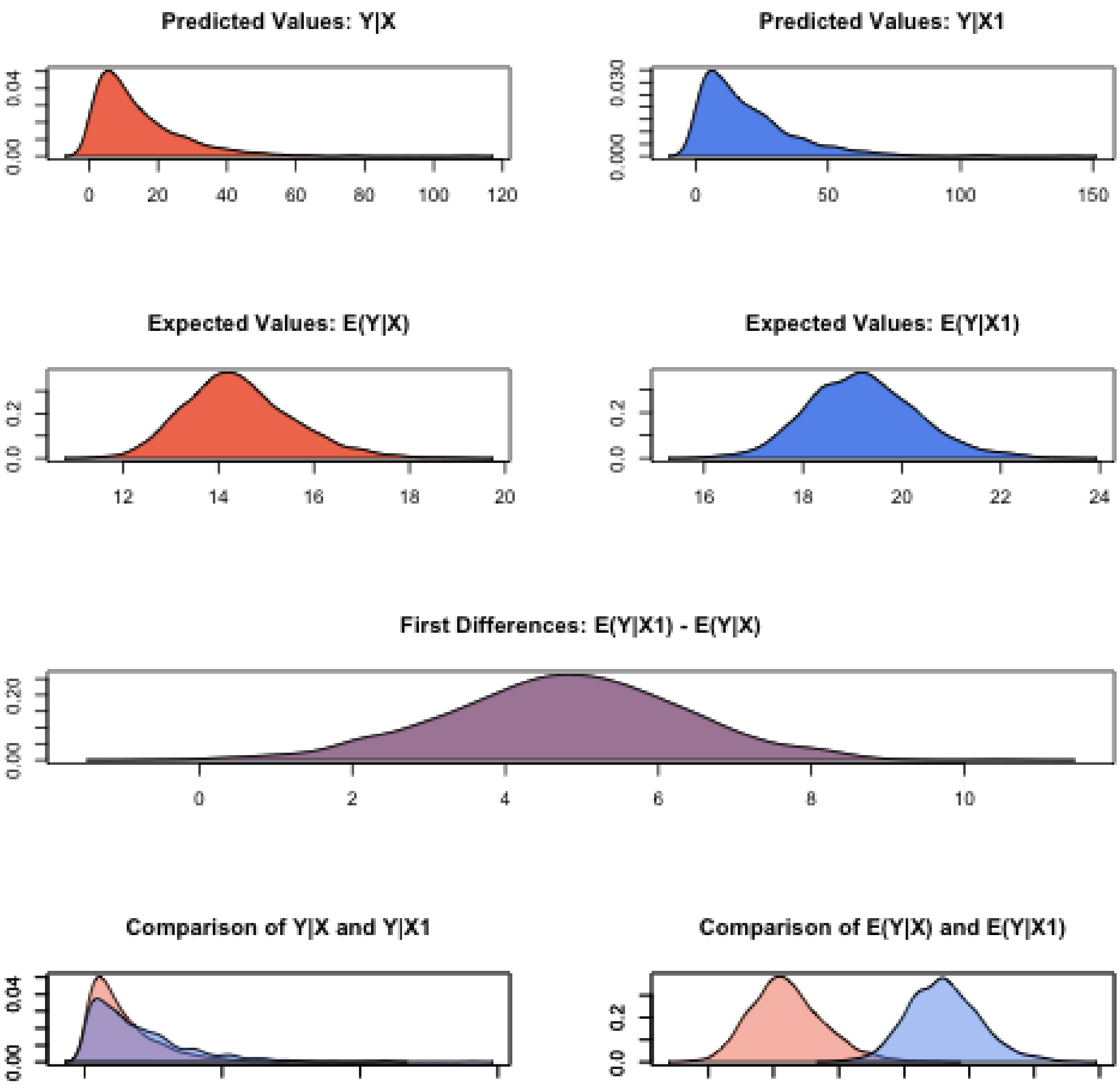


Figure 2.2: Zelig-gamma

where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. Variation in the simulations are due to uncertainty in simulating $E[Y_i(t_i = 0)]$, the counterfactual expected value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.

- In conditional prediction models, the average predicted treatment effect (att.pr) for the treatment group is

$$\frac{1}{\sum_{i=1}^{n} t_i} \sum_{i:t_i=1}^{n} \left\{ Y_i(t_i = 1) - Y_i(\widehat{t_i = 0}) \right\},$$

where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. Variation in the simulations are due to uncertainty in simulating $Y_i(\widehat{t_i = 0})$, the counterfactual predicted value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.

2.3.5 Output Values

The output of each Zelig command contains useful information which you may view. For example, if you run `z.out <- zelig(y ~ x, model = gamma, data)`, then you may examine the available information in `z.out` by using `names(z.out)`, see the coefficients by using z.out$coefficients, and a default summary of information through `summary(z.out)`.

2.3.6 See also

The gamma model is part of the stats package. Advanced users may wish to refer to `help(glm)` and `help(family)`.

2.4 zelig-logit

Logistic Regression for Dichotomous Dependent Variables

Logistic regression specifies a dichotomous dependent variable as a function of a set of explanatory variables.

2.4.1 Syntax

With reference classes:

```
z5 <- zlogit$new()
z5$zelig(Y ~ X1 + X ~ X, data = mydata)
z5$setx()
z5$sim()
```

With the Zelig 4 compatibility wrappers:

```
z.out <- zelig(Y ~ X1 + X2, model = "logit", data = mydata)
x.out <- setx(z.out)
s.out <- sim(z.out, x = x.out, x1 = NULL)
```

2.4.2 Examples

Basic Example

Attaching the sample turnout dataset:

```
data(turnout)
```

Estimating parameter values for the logistic regression:

```
z.out1 <- zelig(vote ~ age + race, model = "logit", data = turnout)

## How to cite this model in Zelig:
##   Kosuke Imai, Gary King, Olivia Lau. 2007.
##   logit: Logistic Regression for Dichotomous Dependent Variables
##   in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
##   http://datascience.iq.harvard.edu/zelig
```

Setting values for the explanatory variables:

```
x.out1 <- setx(z.out1, age = 36, race = "white")
```

Simulating quantities of interest from the posterior distribution.

```
s.out1 <- sim(z.out1, x = x.out1)

summary(s.out1)

##
##  sim x :
##  -----
## ev
##        mean       sd    50%   2.5% 97.5%
## [1,] 0.7481 0.01149 0.7482 0.7264  0.77
## pv
##          0     1
## [1,] 0.265 0.735

plot(s.out1)
```

Simulating First Differences

Estimating the risk difference (and risk ratio) between low education (25th percentile) and high education (75th percentile) while all the other variables held at their default values.

```
z.out2 <- zelig(vote ~ race + educate, model = "logit", data = turnout)

## How to cite this model in Zelig:
##   Kosuke Imai, Gary King, Olivia Lau. 2007.
##   logit: Logistic Regression for Dichotomous Dependent Variables
##   in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
##   http://datascience.iq.harvard.edu/zelig

x.high <- setx(z.out2, educate = quantile(turnout$educate, prob = 0.75))
x.low <- setx(z.out2, educate = quantile(turnout$educate, prob = 0.25))
s.out2 <- sim(z.out2, x = x.high, x1 = x.low)
summary(s.out2)
```

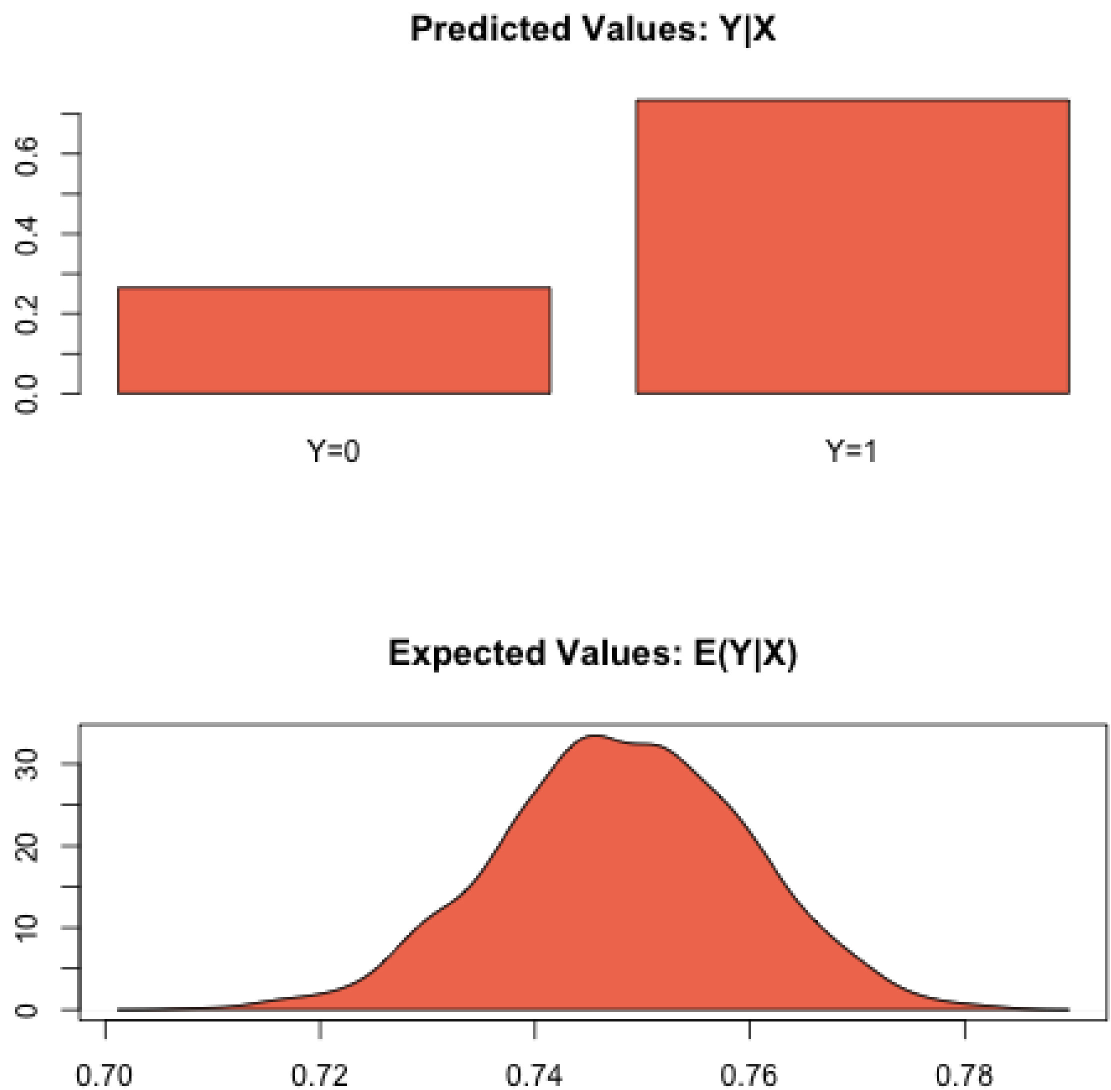


Figure 2.3: Zelig-logit-1

```
##
##  sim x :
##  -----
## ev
##        mean      sd   50%   2.5%  97.5%
## [1,] 0.8227 0.01035 0.823 0.8022 0.8413
## pv
##          0     1
## [1,] 0.166 0.834
##
##  sim x1 :
##  -----
## ev
##        mean      sd    50%   2.5%  97.5%
## [1,] 0.7087 0.01261 0.7086 0.6844 0.7338
## pv
##          0     1
## [1,] 0.282 0.718
## fd
##         mean      sd     50%    2.5%    97.5%
## [1,] -0.1141 0.01133 -0.1139 -0.1362 -0.09217
```

```
plot(s.out2)
```

2.4.3 Model

Let Y_i be the binary dependent variable for observation i which takes the value of either 0 or 1.

- The *stochastic component* is given by

$$\begin{aligned} Y_i \sim & \quad \text{Bernoulli}(y_i \mid \pi_i) \\ = & \quad \pi_i^{y_i}(1-\pi_i)^{1-y_i} \end{aligned}$$

  where $\pi_i = \Pr(Y_i = 1)$.

- The *systematic component* is given by:

$$\pi_i \;=\; \frac{1}{1+\exp(-x_i\beta)}.$$

  where x_i is the vector of k explanatory variables for observation i and β is the vector of coefficients.

2.4.4 Quantities of Interest

- The expected values (qi$ev) for the logit model are simulations of the predicted probability of a success:

$$E(Y) = \pi_i = \frac{1}{1+\exp(-x_i\beta)},$$

  given draws of β from its sampling distribution.

- The predicted values (qi$pr) are draws from the Binomial distribution with mean equal to the simulated expected value π_i.

- The first difference (qi$fd) for the logit model is defined as

$$\text{FD} = \Pr(Y = 1 \mid x_1) - \Pr(Y = 1 \mid x).$$

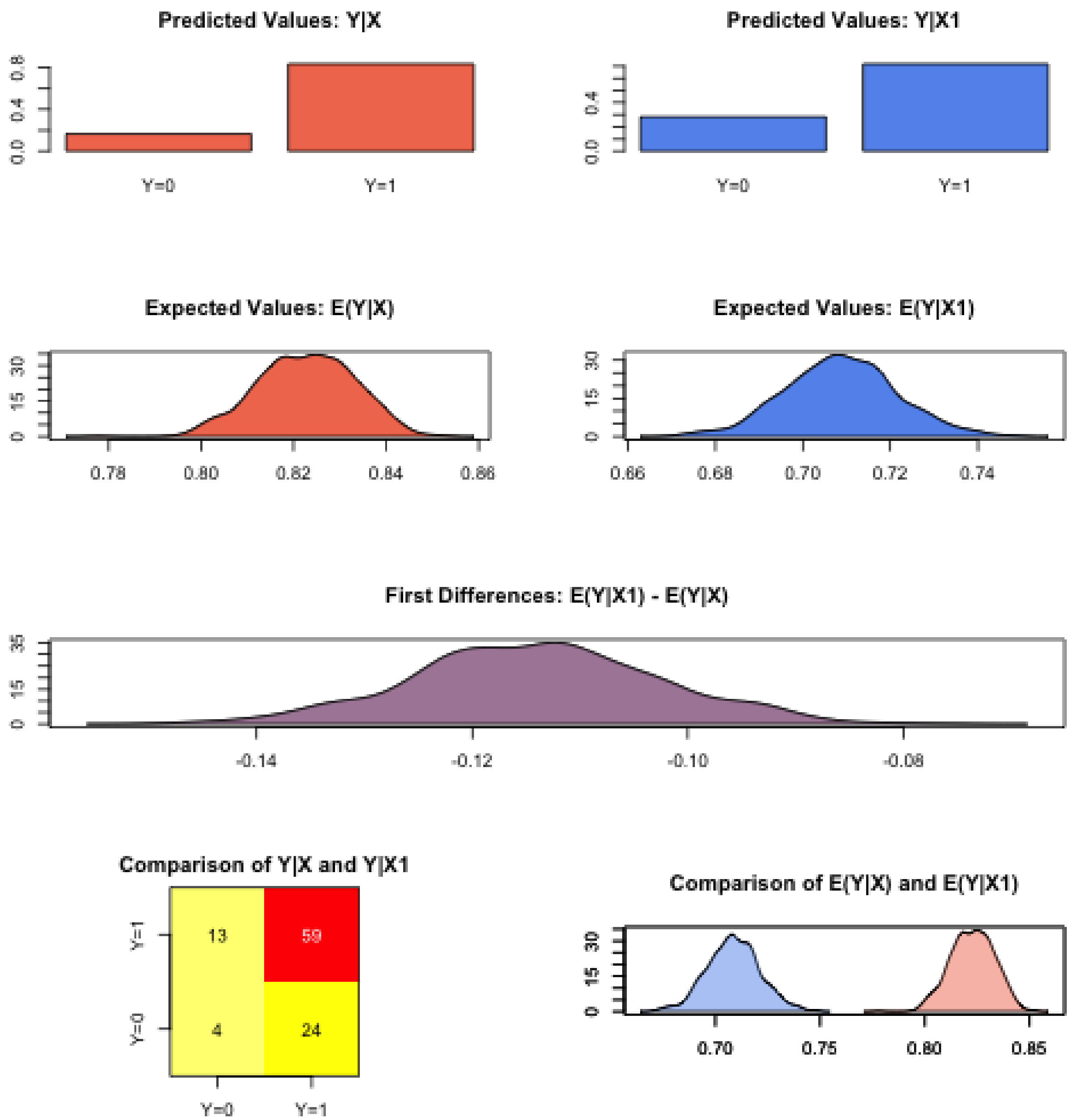


Figure 2.4: Zelig-logit-2

- The risk ratio (qi$rr) is defined as

$$\text{RR} = \Pr(Y = 1 \mid x_1) \;/\; \Pr(Y = 1 \mid x).$$

- In conditional prediction models, the average expected treatment effect (att.ev) for the treatment group is

$$\frac{1}{\sum_{i=1}^n t_i}\sum_{i:t_i=1}^n \left\{ Y_i(t_i = 1) - E[Y_i(t_i = 0)] \right\},$$

  where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. Variation in the simulations are due to uncertainty in simulating $E[Y_i(t_i = 0)]$, the counterfactual expected value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.

- In conditional prediction models, the average predicted treatment effect (att.pr) for the treatment group is

$$\frac{1}{\sum_{i=1}^n t_i}\sum_{i:t_i=1}^n \left\{ Y_i(t_i = 1) - \widehat{Y_i(t_i = 0)} \right\},$$

  where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. Variation in the simulations are due to uncertainty in simulating $\widehat{Y_i(t_i = 0)}$, the counterfactual predicted value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.

2.4.5 Output Values

The output of each Zelig command contains useful information which you may view. For example, if you run `z.out <- zelig(y ~ x, model = logit, data)`, then you may examine the available information in `z.out` by using `names(z.out)`, see the coefficients by using z.out$coefficients, and a default summary of information through `summary(z.out)`.

2.4.6 See also

The logit model is part of the stats package. Advanced users may wish to refer to `help(glm)` and `help(family)`.

2.5 zelig-lognorm

Log-Normal Regression for Duration Dependent Variables

The log-normal model describes an event's duration, the dependent variable, as a function of a set of explanatory variables. The log-normal model may take time censored dependent variables, and allows the hazard rate to increase and decrease.

2.5.1 Syntax

With reference classes:

```
z5 <- zlognorm$new()
z5$zelig(Surv(Y, C) ~ X, data = mydata)
z5$setx()
z5$sim()
```

With reference classes:

```
z5 <- zlognorm$new()
z5$zelig(Surv(Y, C) ~ X, data = mydata)
z5$setx()
z5$sim()
```

With the Zelig 4 compatibility wrappers:

```
z.out <- zelig(Surv(Y, C) ~ X, model = "lognorm", data = mydata)
x.out <- setx(z.out)
s.out <- sim(z.out, x = x.out)
```

Log-normal models require that the dependent variable be in the form Surv(Y, C), where Y and C are vectors of length n. For each observation i in 1, $\ldots$, n, the value y_i is the duration (lifetime, for example) of each subject, and the associated c_i is a binary variable such that $c_i = 1$ if the duration is not censored (*e.g.*, the subject dies during the study) or $c_i = 0$ if the duration is censored (*e.g.*, the subject is still alive at the end of the study). If c_i is omitted, all Y are assumed to be completed; that is, time defaults to 1 for all observations.

2.5.2 Input Values

In addition to the standard inputs, zelig() takes the following additional options for lognormal regression:

- robust: defaults to FALSE. If TRUE, zelig() computes robust standard errors based on sandwich estimators (see and) based on the options in cluster.
- cluster: if robust = TRUE, you may select a variable to define groups of correlated observations. Let x3 be a variable that consists of either discrete numeric values, character strings, or factors that define strata. Then

```
z.out <- zelig(y ~ x1 + x2, robust = TRUE, cluster = "x3", model = "exp", data = mydata)
```

means that the observations can be correlated within the strata defined by the variable x3, and that robust standard errors should be calculated according to those clusters. If robust = TRUE but cluster is not specified, zelig() assumes that each observation falls into its own cluster.

2.5.3 Example

Attach the sample data:

```
data(coalition)
```

Estimate the model:

```
z.out <- zelig(Surv(duration, ciep12) ~ fract + numst2, model ="lognorm",  data = coalition)

## How to cite this model in Zelig:
##   Matthew Owen, Olivia Lau, Kosuke Imai, Gary King. 2007.
##   lognorm: Log-Normal Regression for Duration Dependent Variables
##   in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
##   http://datascience.iq.harvard.edu/zelig
```

View the regression output:

```
summary(z.out)
```

```
## Model: 1Call:
## survival::survreg(formula = Surv(duration, ciep12) ~ fract +
##     numst2, data = ., dist = "lognormal", model = FALSE)
##
## Coefficients:
## (Intercept)       fract      numst2
##    5.366670   -0.004438    0.559833
##
## Scale= 1.2
##
## Loglik(model)= -1078   Loglik(intercept only)= -1101
##  Chisq= 46.58 on 2 degrees of freedom, p= 7.7e-11
## n= 314
## Next step: Use 'setx' method
```

Set the baseline values (with the ruling coalition in the minority) and the alternative values (with the ruling coalition in the majority) for X:

```
x.low <- setx(z.out, numst2 = 0)
x.high <- setx(z.out, numst2= 1)
```

Simulate expected values (qi$ev) and first differences (qi$fd):

```
s.out <- sim(z.out, x = x.low, x1 = x.high)
```

```
summary(s.out)
```

```
##
##  sim x :
##  -----
## ev
##    mean   sd   50%  2.5% 97.5%
## 1 18.4 2.41 18.37 14.05 23.34
## pv
##    mean   sd   50%  2.5% 97.5%
## 1 18.4 2.41 18.37 14.05 23.34
##
##  sim x1 :
##  -----
## ev
##     mean    sd   50% 2.5% 97.5%
## 1 32.26 3.699 31.99 25.9 40.29
## pv
##     mean    sd   50% 2.5% 97.5%
## 1 32.26 3.699 31.99 25.9 40.29
## fd
##     mean    sd   50%  2.5% 97.5%
## 1 13.86 3.612 13.76 6.925 21.12
```

```
plot(s.out)
```

2.5.4 Model

Let Y_i^* be the survival time for observation i with the density function $f(y)$ and the corresponding distribution function $F(t) = \int_0^t f(y)dy$. This variable might be censored for some observations at a fixed time y_c such that the fully

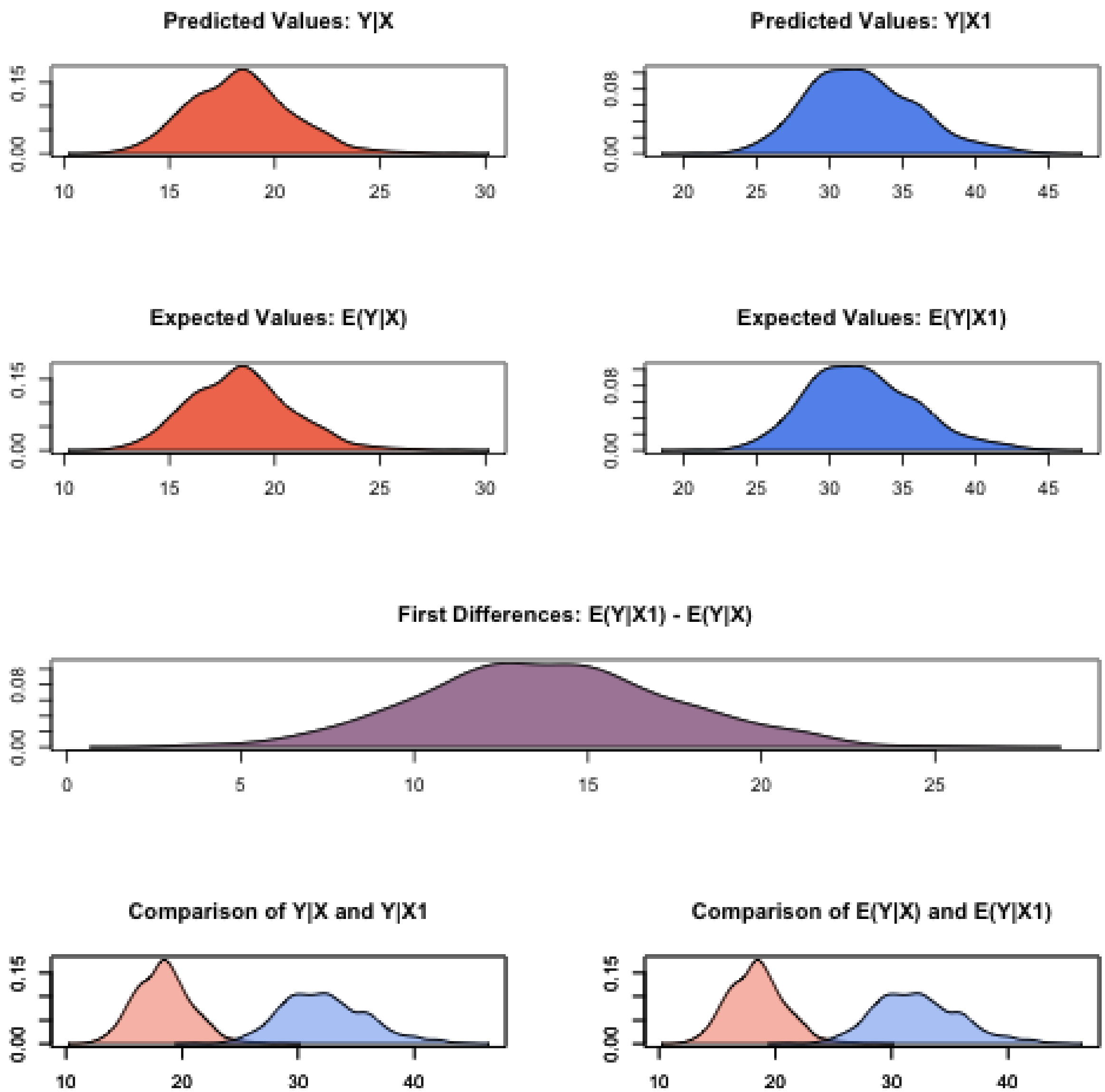


Figure 2.5: Zelig-lognorm

observed dependent variable, Y_i, is defined as

$$Y_i = \begin{cases} Y_i^* & \text{if } Y_i^* \leq y_c \\ y_c & \text{if } Y_i^* > y_c \end{cases}$$

- The *stochastic component* is described by the distribution of the partially observed variable, Y^*. For the lognormal model, there are two equivalent representations:

$$Y_i^* \sim \text{LogNormal}(\mu_i, \sigma^2) \text{ or } \quad \log(Y_i^*) \sim \text{Normal}(\mu_i, \sigma^2)$$

  where the parameters μ_i and σ^2 are the mean and variance of the Normal distribution. (Note that the output from zelig() parameterizes scale:math:‘ = sigma‘.)

  In addition, survival models like the lognormal have three additional properties. The hazard function $h(t)$ measures the probability of not surviving past time t given survival up to t. In general, the hazard function is equal to $f(t)/S(t)$ where the survival function $S(t) = 1 - \int_0^t f(s)ds$ represents the fraction still surviving at time t. The cumulative hazard function $H(t)$ describes the probability of dying before time t. In general, $H(t) = \int_0^t h(s)ds = -\log S(t)$. In the case of the lognormal model,

$$\begin{aligned} h(t) = & \quad \frac{1}{\sqrt{2\pi}\,\sigma t\, S(t)} \exp\left\{-\frac{1}{2\sigma^2}(\log \lambda t)^2\right\} \\ S(t) = & \quad 1 - \Phi\left(\frac{1}{\sigma}\log \lambda t\right) \\ H(t) = & \quad -\log\left\{1 - \Phi\left(\frac{1}{\sigma}\log \lambda t\right)\right\} \end{aligned}$$

  where $\Phi(\cdot)$ is the cumulative density function for the Normal distribution.

- The *systematic component* is described as:

$$\mu_i = x_i\beta.$$

2.5.5 Quantities of Interest

- The expected values (qi$ev) for the lognormal model are simulations of the expected duration:

$$E(Y) = \exp\left(\mu_i + \frac{1}{2}\sigma^2\right),$$

  given draws of β and σ from their sampling distributions.

- The predicted value is a draw from the log-normal distribution given simulations of the parameters (λ_i, σ).
- The first difference (qi$fd) is

$$\text{FD} = E(Y \mid x_1) - E(Y \mid x).$$

- In conditional prediction models, the average expected treatment effect (att.ev) for the treatment group is

$$\frac{1}{\sum_{i=1}^n t_i} \sum_{i:t_i=1}^n \{Y_i(t_i = 1) - E[Y_i(t_i = 0)]\},$$

  where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. When $Y_i(t_i = 1)$ is censored rather than observed, we replace it with a simulation from the model given available knowledge of the censoring process. Variation in the simulations is due to two factors: uncertainty in the imputation process for censored y_i^* and uncertainty in simulating $E[Y_i(t_i = 0)]$, the counterfactual expected value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.

- In conditional prediction models, the average predicted treatment effect (att.pr) for the treatment group is

$$\frac{1}{\sum_{i=1}^{n} t_i} \sum_{i:t_i=1}^{n} \{Y_i(t_i = 1) - \widehat{Y_i(t_i = 0)}\},$$

  where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. When $Y_i(t_i = 1)$ is censored rather than observed, we replace it with a simulation from the model given available knowledge of the censoring process. Variation in the simulations are due to two factors: uncertainty in the imputation process for censored y_i^* and uncertainty in simulating $\widehat{Y_i(t_i = 0)}$, the counterfactual predicted value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.

2.5.6 Output Values

The output of each Zelig command contains useful information which you may view. For example, if you run `z.out <- zelig(Surv(Y, C) ~ X, model = lognorm, data)`, then you may examine the available information in `z.out` by using `names(z.out)`, see the coefficients by using z.out$coefficients, and a default summary of information through `summary(z.out)`.

2.5.7 See also

The exponential function is part of the survival library by by Terry Therneau, ported to R by Thomas Lumley. Advanced users may wish to refer to `help(survfit)` in the survival library.

2.6 zelig-ls

Least Squares Regression for Continuous Dependent Variables

Use least squares regression analysis to estimate the best linear predictor for the specified dependent variables.

2.6.1 Syntax

With reference classes:

```
z5 <- zls$new()
z5$zelig(Y ~ X1 + X ~ X, data = mydata)
z5$setx()
z5$sim()
```

With the Zelig 4 compatibility wrappers:

```
z.out <- zelig(Y ~ X1 + X2, model = "ls", data = mydata)
x.out <- setx(z.out)
s.out <- sim(z.out, x = x.out)
```

2.6.2 Examples

Basic Example with First Differences

Attach sample data:

```
data(macro)
```

Estimate model:

```
z.out1 <- zelig(unem ~ gdp + capmob + trade, model = "ls", data = macro)

## How to cite this model in Zelig:
##   Kosuke Imai, Gary King, and Olivia Lau. 2007.
##   ls: Least Squares Regression for Continuous Dependent Variables
##   in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
##   http://datascience.iq.harvard.edu/zelig
```

Summarize regression coefficients:

```
summary(z.out1)

## Model: 1
## Call:
## stats::lm(formula = unem ~ gdp + capmob + trade, data = .)
##
## Coefficients:
## (Intercept)          gdp       capmob        trade
##      6.1813      -0.3236       1.4219       0.0199
##
## Next step: Use 'setx' method
```

Set explanatory variables to their default (mean/mode) values, with high (80th percentile) and low (20th percentile) values for the trade variable:

```
x.high <- setx(z.out1, trade = quantile(macro$trade, 0.8))
x.low <- setx(z.out1, trade = quantile(macro$trade, 0.2))
```

Generate first differences for the effect of high versus low trade on GDP:

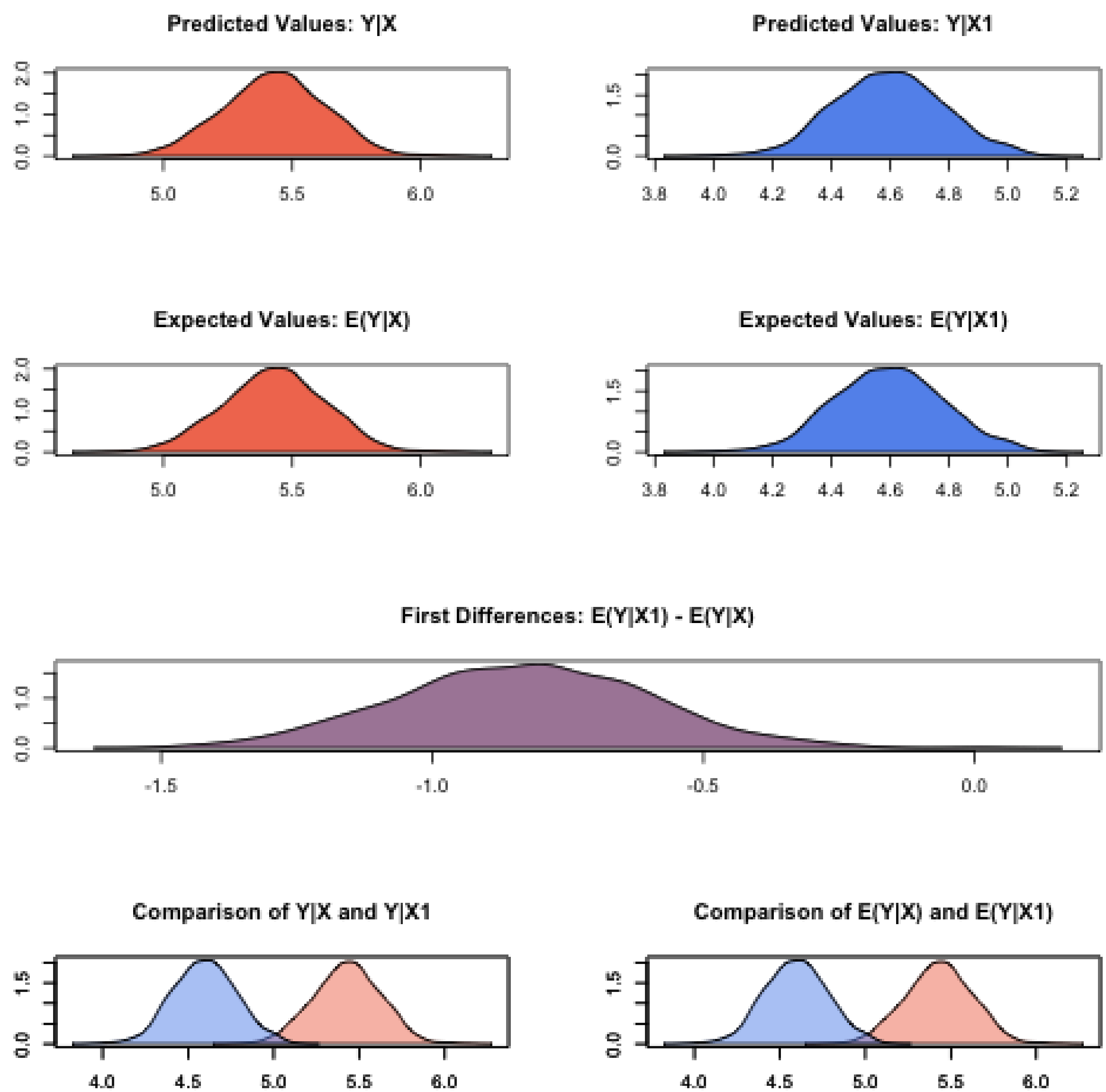
```
s.out1 <- sim(z.out1, x = x.high, x1 = x.low)

summary(s.out1)

##
##  sim x :
##  -----
## ev
##    mean     sd   50%  2.5% 97.5%
## 1 5.431 0.1964 5.432 5.056 5.796
## pv
##    mean     sd   50%  2.5% 97.5%
## 1 5.431 0.1964 5.432 5.056 5.796
##
##  sim x1 :
##  -----
## ev
##    mean     sd 50%  2.5% 97.5%
## 1 4.597 0.1838 4.6 4.245 4.969
```

```
## pv
##    mean     sd 50%  2.5% 97.5%
## 1 4.597 0.1838 4.6 4.245 4.969
## fd
##      mean     sd     50%   2.5%   97.5%
## 1 -0.8342 0.2281 -0.8266 -1.278 -0.3786
```

```
plot(s.out1)
```



Using Dummy Variables

Estimate a model with fixed effects for each country (see for help with dummy variables). Note that you do not need to create dummy variables, as the program will automatically parse the unique values in the selected variable into discrete levels.

```
z.out2 <- zelig(unem ~ gdp + trade + capmob + as.factor(country), model = "ls", data = macro)

## How to cite this model in Zelig:
##   Kosuke Imai, Gary King, and Olivia Lau. 2007.
##   ls: Least Squares Regression for Continuous Dependent Variables
##   in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
##   http://datascience.iq.harvard.edu/zelig
```

Set values for the explanatory variables, using the default mean/mode values, with country set to the United States and Japan, respectively:

```
x.US <- setx(z.out2, country = "United States")
x.Japan <- setx(z.out2, country = "Japan")
```

Simulate quantities of interest:

```
s.out2 <- sim(z.out2, x = x.US, x1 = x.Japan)

plot(s.out2)
```

2.6.3 Model

- The *stochastic component* is described by a density with mean μ_i and the common variance σ^2

$$Y_i \;\sim\; f(y_i \mid \mu_i, \sigma^2).$$

- The *systematic component* models the conditional mean as

$$\mu_i = x_i\beta$$

  where x_i is the vector of covariates, and β is the vector of coefficients.

  The least squares estimator is the best linear predictor of a dependent variable given x_i, and minimizes the sum of squared residuals, $\sum_{i=1}^{n}(Y_i - x_i\beta)^2$.

2.6.4 Quantities of Interest

- The expected value (qi$ev) is the mean of simulations from the stochastic component,

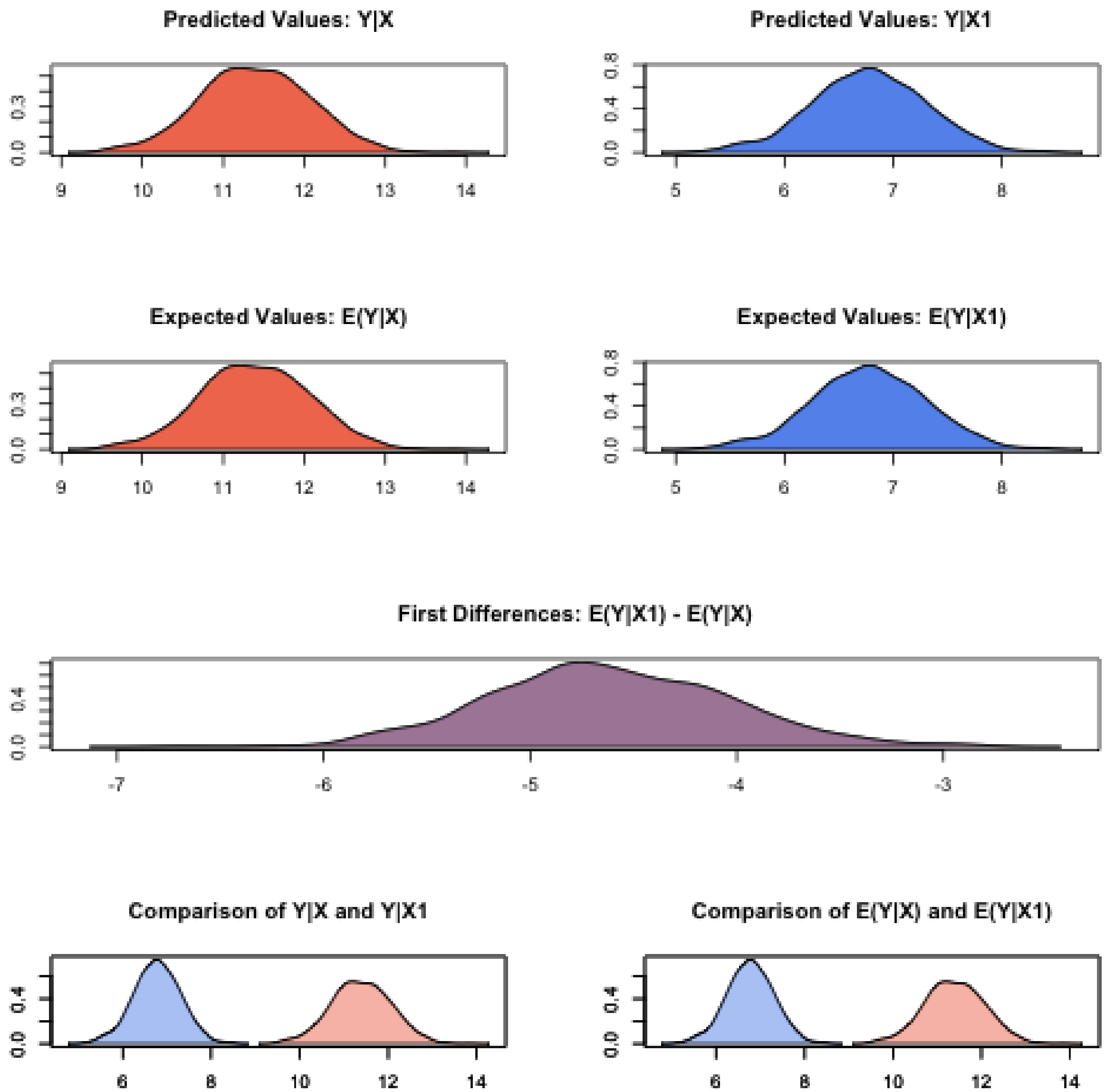
$$E(Y) = x_i\beta,$$

  given a draw of β from its sampling distribution.

- In conditional prediction models, the average expected treatment effect (att.ev) for the treatment group is

$$\frac{1}{\sum_{i=1}^{n} t_i} \sum_{i:t_i=1}^{n} \left\{ Y_i(t_i = 1) - E[Y_i(t_i = 0)] \right\},$$

  where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. Variation in the simulations are due to uncertainty in simulating $E[Y_i(t_i = 0)]$, the counterfactual expected value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.



Predicted Values: Y|X
Predicted Values: Y|X1
Expected Values: E(Y|X)
Expected Values: E(Y|X1)
First Differences: E(Y|X1) - E(Y|X)
Comparison of Y|X and Y|X1
Comparison of E(Y|X) and E(Y|X1)

2.6.5 Output Values

The output of each Zelig command contains useful information which you may view. For example, if you run `z.out <- zelig(y ~ x, model = ls, data)`, then you may examine the available information in `z.out` by using `names(z.out)`, see the coefficients by using z.out$coefficients, and a default summary of information through `summary(z.out)`. Other elements available through the $ operator are listed below.

- From the zelig() output object z.out, you may extract:
  - coefficients: parameter estimates for the explanatory variables.
  - residuals: the working residuals in the final iteration of the IWLS fit.
  - fitted.values: fitted values.
  - df.residual: the residual degrees of freedom.
  - zelig.data: the input data frame if save.data = TRUE.
- From summary(z.out), you may extract:
  - coefficients: the parameter estimates with their associated standard errors, p-values, and t-statistics.

$$\hat{\beta} = \left(\sum_{i=1}^{n} x_i' x_i\right)^{-1} \sum x_i y_i$$

  - sigma: the square root of the estimate variance of the random error e:

$$\hat{\sigma} = \frac{\sum (Y_i - x_i \hat{\beta})^2}{n-k}$$

  - r.squared: the fraction of the variance explained by the model.

$$R^2 = 1 - \frac{\sum (Y_i - x_i \hat{\beta})^2}{\sum (y_i - \bar{y})^2}$$

  - adj.r.squared: the above R^2 statistic, penalizing for an increased number of explanatory variables.
  - cov.unscaled: a $k \times k$ matrix of unscaled covariances.

2.6.6 See also

The least squares regression is part of the stats package by William N. Venables and Brian D. Ripley .In addition, advanced users may wish to refer to `help(lm)` and `help(lm.fit)`.

2.7 zelig-negbin

Negative Binomial Regression for Event Count Dependent Variables

Use the negative binomial regression if you have a count of events for each observation of your dependent variable. The negative binomial model is frequently used to estimate over-dispersed event count models.

2.7.1 Syntax

With reference classes:

```
z5 <- znegbin$new()
z5$zelig(Y ~ X1 + X ~ X, data = mydata)
z5$setx()
z5$sim()
```

With the Zelig 4 compatibility wrappers:

```
z.out <- zelig(Y ~ X1 + X2, model = "negbin", data = mydata)
x.out <- setx(z.out)
s.out <- sim(z.out, x = x.out)
```

2.7.2 Example

Load sample data:

```
data(sanction)
```

Estimate the model:

```
z.out <- zelig(num ~ target + coop, model = "negbinom", data = sanction)
```

```
## Error: Model 'negbinom' not found
```

```
summary(z.out)
```

```
## Model: 1Call:
## survival::survreg(formula = Surv(duration, ciep12) ~ fract +
##     numst2, data = ., dist = "lognormal", model = FALSE)
##
## Coefficients:
## (Intercept)       fract      numst2
##    5.366670   -0.004438    0.559833
##
## Scale= 1.2
##
## Loglik(model)= -1078   Loglik(intercept only)= -1101
##  Chisq= 46.58 on 2 degrees of freedom, p= 7.7e-11
## n= 314
## Next step: Use 'setx' method
```

Set values for the explanatory variables to their default mean values:

```
x.out <- setx(z.out)
```

Simulate fitted values:

```
s.out <- sim(z.out, x = x.out)
```

```
summary(s.out)
```

```
##
##  sim x :
##  -----
## ev
```

```
##    mean    sd   50%  2.5% 97.5%
## 1 26.05 2.643 25.86 21.33  31.8
## pv
##    mean    sd   50%  2.5% 97.5%
## 1 26.05 2.643 25.86 21.33  31.8

plot(s.out)
```

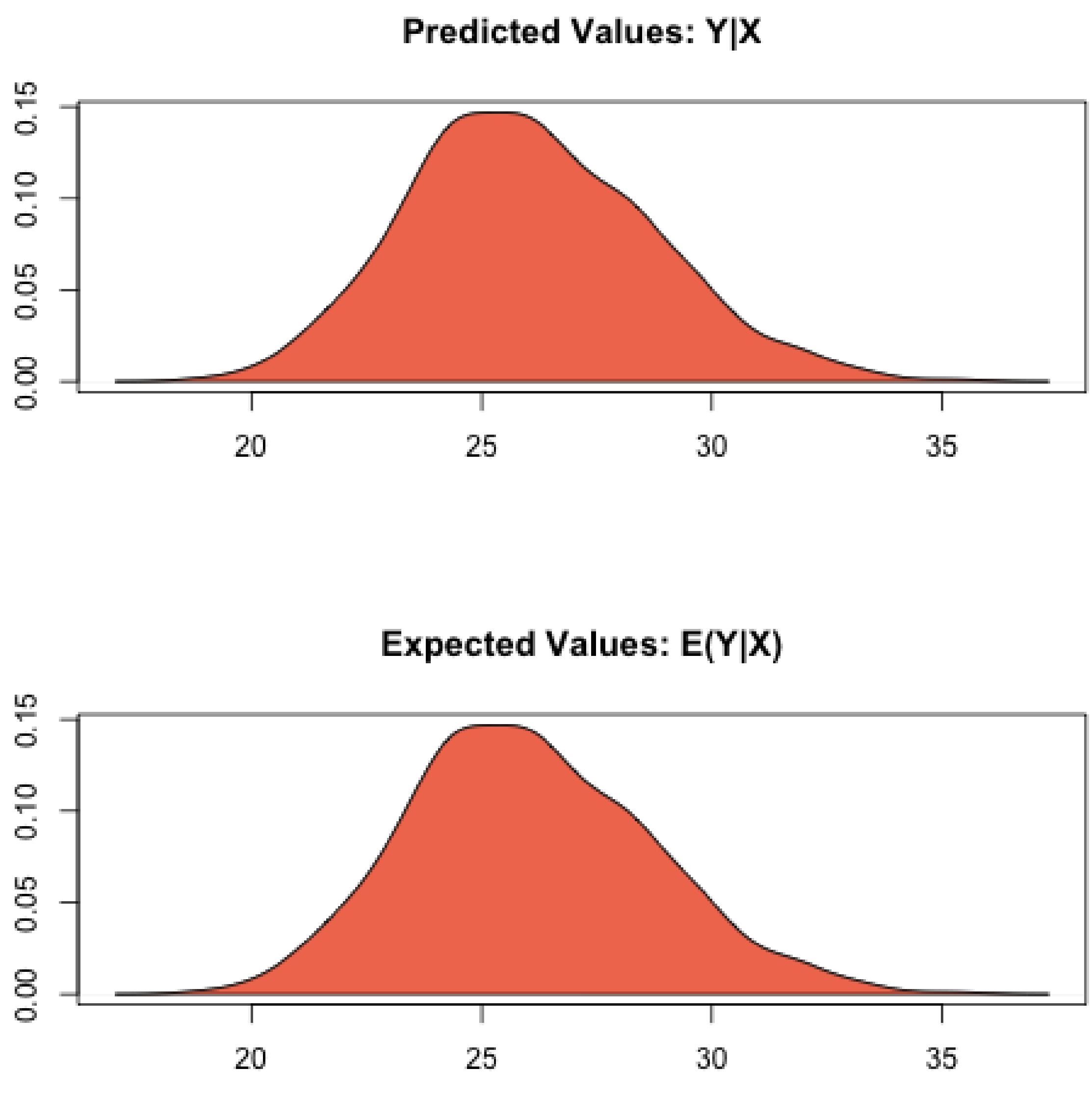


Figure 2.6: Zelig-negbin

2.7.3 Model

Let Y_i be the number of independent events that occur during a fixed time period. This variable can take any non-negative integer value.

- The negative binomial distribution is derived by letting the mean of the Poisson distribution vary according to a fixed parameter ζ given by the Gamma distribution. The *stochastic component* is given by

$$\begin{aligned} Y_i \mid \zeta_i &\sim \text{Poisson}(\zeta_i \mu_i), \\ \zeta_i &\sim \frac{1}{\theta}\text{Gamma}(\theta). \end{aligned}$$

  The marginal distribution of Y_i is then the negative binomial with mean μ_i and variance $\mu_i + \mu_i^2/\theta$:

$$\begin{aligned} Y_i &\sim \text{NegBin}(\mu_i, \theta), \\ &= \frac{\Gamma(\theta + y_i)}{y!\,\Gamma(\theta)} \frac{\mu_i^{y_i}\,\theta^{\theta}}{(\mu_i + \theta)^{\theta + y_i}}, \end{aligned}$$

  where θ is the systematic parameter of the Gamma distribution modeling ζ_i.

- The *systematic component* is given by

$$\mu_i = \exp(x_i \beta)$$

  where x_i is the vector of k explanatory variables and β is the vector of coefficients.

2.7.4 Quantities of Interest

- The expected values (qi$ev) are simulations of the mean of the stochastic component. Thus,

$$E(Y) = \mu_i = \exp(x_i \beta),$$

  given simulations of β.

- The predicted value (qi$pr) drawn from the distribution defined by the set of parameters (μ_i, θ).
- The first difference (qi$fd) is

$$\text{FD} \; = \; E(Y|x_1) - E(Y \mid x)$$

- In conditional prediction models, the average expected treatment effect (att.ev) for the treatment group is

$$\frac{1}{\sum_{i=1}^n t_i} \sum_{i:t_i=1}^n \left\{ Y_i(t_i = 1) - E[Y_i(t_i = 0)] \right\},$$

  where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. Variation in the simulations are due to uncertainty in simulating $E[Y_i(t_i = 0)]$, the counterfactual expected value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.

- In conditional prediction models, the average predicted treatment effect (att.pr) for the treatment group is

$$\frac{1}{\sum_{i=1}^n t_i} \sum_{i:t_i=1}^n \left\{ Y_i(t_i = 1) - Y_i(\widehat{t_i = 0}) \right\},$$

  where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. Variation in the simulations are due to uncertainty in simulating $Y_i(\widehat{t_i = 0})$, the counterfactual predicted value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.

2.7.5 Output Values

The output of each Zelig command contains useful information which you may view. For example, if you run `z.out <- zelig(y ~ x, model = negbin, data)`, then you may examine the available information in `z.out` by using `names(z.out)`, see the coefficients by using z.out$coefficients, and a default summary of information through `summary(z.out)`.

2.7.6 See also

The negative binomial model is part of the MASS package by William N. Venable and Brian D. Ripley . Advanced users may wish to refer to ‘‘help(glm.nb)‘.

2.8 zelig-normal

Normal Regression for Continuous Dependent Variables

The Normal regression model is a close variant of the more standard least squares regression model (see). Both models specify a continuous dependent variable as a linear function of a set of explanatory variables. The Normal model reports maximum likelihood (rather than least squares) estimates. The two models differ only in their estimate for the stochastic parameter σ.

2.8.1 Syntax

With reference classes:

```
z5 <- znormal$new()
z5$zelig(Y ~ X1 + X ~ X, data = mydata)
z5$setx()
z5$sim()
```

With the Zelig 4 compatibility wrappers:

```
z.out <- zelig(Y ~ X1 + X2, model = "normal", data = mydata)
x.out <- setx(z.out)
s.out <- sim(z.out, x = x.out)
```

2.8.2 Examples

Basic Example with First Differences

Attach sample data:

```
data(macro)
```

Estimate model:

```
z.out1 <- zelig(unem ~ gdp + capmob + trade, model = "normal", data = macro)
```

```
## How to cite this model in Zelig:
##   Kosuke Imai, Gary King, Olivia Lau. 2008.
##   normal: Normal Regression for Continuous Dependent Variables
##   in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
##   http://datascience.iq.harvard.edu/zelig
```

Summarize of regression coefficients:

```
summary(z.out1)
```

```
## Model: 1
## Call:  stats::glm(formula = unem ~ gdp + capmob + trade, family = gaussian("identity"),
##     data = .)
##
## Coefficients:
## (Intercept)          gdp       capmob        trade
##      6.1813      -0.3236       1.4219       0.0199
##
## Degrees of Freedom: 349 Total (i.e. Null);  346 Residual
## Null Deviance:       3660
## Residual Deviance: 2610  AIC: 1710
## Next step: Use 'setx' method
```

Set explanatory variables to their default (mean/mode) values, with high (80th percentile) and low (20th percentile) values for trade:

```
x.high <- setx(z.out1, trade = quantile(macro$trade, 0.8))
x.low <- setx(z.out1, trade = quantile(macro$trade, 0.2))
```

Generate first differences for the effect of high versus low trade on GDP:

```
s.out1 <- sim(z.out1, x = x.high, x1 = x.low)
```

```
summary(s.out1)
```

```
##
##  sim x :
##  -----
## ev
##       mean     sd   50%  2.5% 97.5%
## [1,] 5.427 0.1905 5.426 5.066  5.78
## pv
##       mean    sd 50%   2.5% 97.5%
## [1,] 5.491 2.833 5.4 0.3017 11.19
##
##  sim x1 :
##  -----
## ev
##       mean     sd   50%  2.5% 97.5%
## [1,] 4.597 0.1845 4.599 4.238 4.979
## pv
##       mean    sd   50%   2.5% 97.5%
## [1,] 4.449 2.687 4.432 -1.007 9.682
## fd
##       mean     sd     50%   2.5%   97.5%
## [1,] -0.83 0.2307 -0.8362 -1.238 -0.3597
```

A visual summary of quantities of interest:

```
plot(s.out1)
```

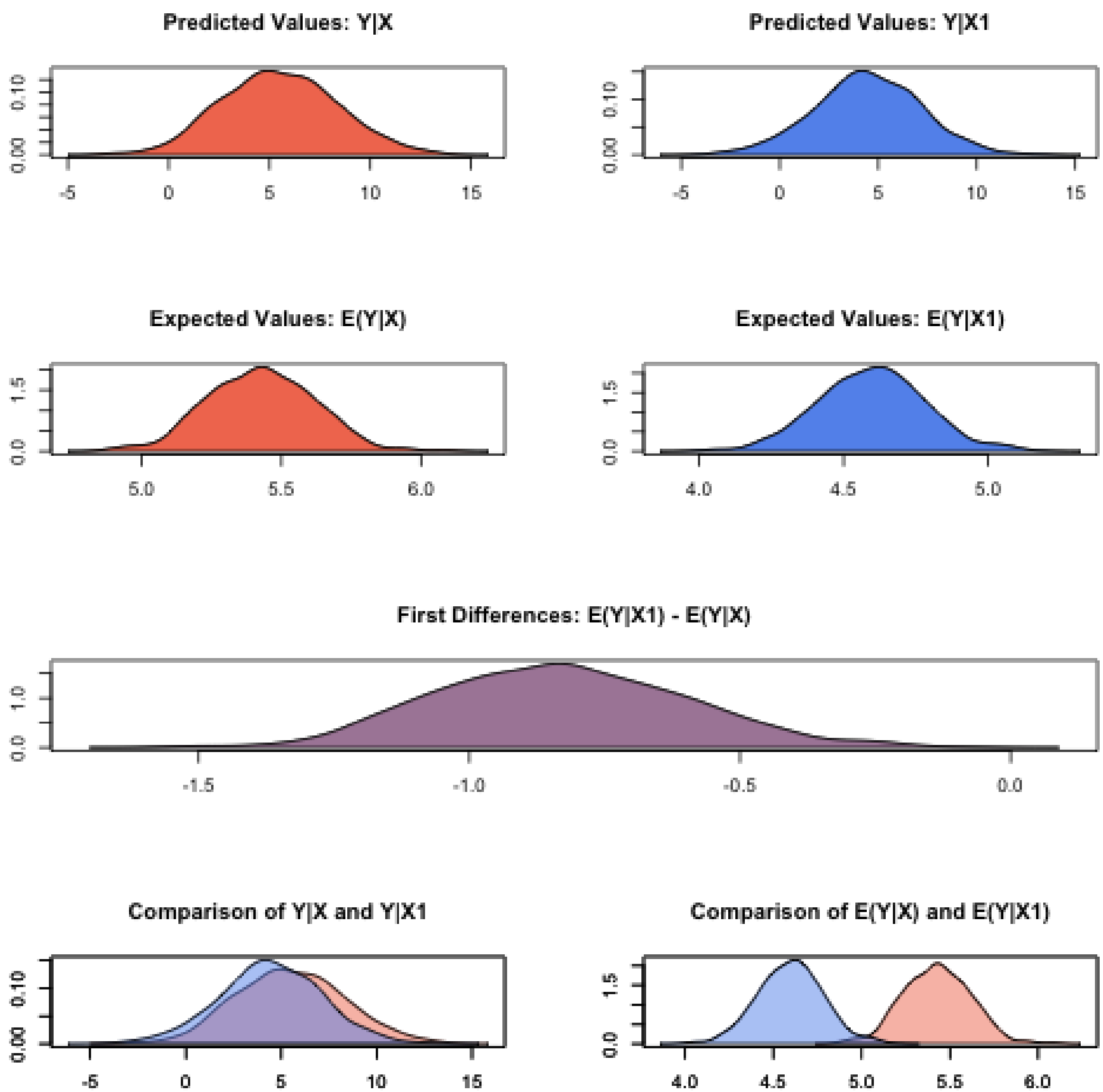


Figure 2.7: Zelig-normal

2.8.3 Model

Let Y_i be the continuous dependent variable for observation i.

- The *stochastic component* is described by a univariate normal model with a vector of means μ_i and scalar variance σ^2:

$$Y_i \sim \text{Normal}(\mu_i, \sigma^2).$$

- The *systematic component* is

$$\mu_i \;=\; x_i\beta,$$

where x_i is the vector of k explanatory variables and β is the vector of coefficients.

2.8.4 Quantities of Interest

- The expected value (qi$ev) is the mean of simulations from the the stochastic component,

$$E(Y) = \mu_i = x_i\beta,$$

given a draw of β from its posterior.

- The predicted value (qi$pr) is drawn from the distribution defined by the set of parameters (μ_i, σ).
- The first difference (qi$fd) is:

$$\text{FD} \;=\; E(Y \mid x_1) - E(Y \mid x)$$

- In conditional prediction models, the average expected treatment effect (att.ev) for the treatment group is

$$\frac{1}{\sum_{i=1}^{n} t_i} \sum_{i:t_i=1}^{n} \left\{ Y_i(t_i = 1) - E[Y_i(t_i = 0)] \right\},$$

where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. Variation in the simulations are due to uncertainty in simulating $E[Y_i(t_i = 0)]$, the counterfactual expected value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.

- In conditional prediction models, the average predicted treatment effect (att.pr) for the treatment group is

$$\frac{1}{\sum_{i=1}^{n} t_i} \sum_{i:t_i=1}^{n} \left\{ Y_i(t_i = 1) - Y_i(\widehat{t_i = 0}) \right\},$$

where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. Variation in the simulations are due to uncertainty in simulating $Y_i(\widehat{t_i = 0})$, the counterfactual predicted value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.

2.8.5 Output Values

The output of each Zelig command contains useful information which you may view. For example, if you run `z.out <- zelig(y ~ x, model = normal, data)`, then you may examine the available information in `z.out` by using `names(z.out)`, see the coefficients by using z.out$coefficients, and a default summary of information through `summary(z.out)`.

2.8.6 See also

The normal model is part of the stats package by . Advanced users may wish to refer to `help(glm)` and `help(family)`.

2.9 zelig-poisson

Poisson Regression for Event Count Dependent Variables

Use the Poisson regression model if the observations of your dependent variable represents the number of independent events that occur during a fixed period of time (see the negative binomial model, , for over-dispersed event counts.) For a Bayesian implementation of this model, see .

2.9.1 Syntax

With reference classes:

```
z5 <- zpoisson$new()
z5$zelig(Y ~ X1 + X ~ X, data = mydata)
z5$setx()
z5$sim()
```

With the Zelig 4 compatibility wrappers:

```
z.out <- zelig(Y ~ X1 + X2, model = "poisson", data = mydata)
x.out <- setx(z.out)
s.out <- sim(z.out, x = x.out)
```

2.9.2 Example

Load sample data:

```
data(sanction)
```

Estimate Poisson model:

```
z.out <- zelig(num ~ target + coop, model = "poisson", data = sanction)

## How to cite this model in Zelig:
##   Kosuke Imai, Gary King, Olivia Lau. 2007.
##   poisson: Poisson Regression for Event Count Dependent Variables
##   in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
##   http://datascience.iq.harvard.edu/zelig

summary(z.out)

## Model: 1
## Call:  stats::glm(formula = num ~ target + coop, family = poisson("log"),
##     data = .)
##
## Coefficients:
## (Intercept)       target         coop
##      -0.968       -0.021        1.211
##
## Degrees of Freedom: 77 Total (i.e. Null);  75 Residual
## Null Deviance:       1580
## Residual Deviance: 721    AIC: 944
## Next step: Use 'setx' method
```

Set values for the explanatory variables to their default mean values:

```
x.out <- setx(z.out)
```

Simulate fitted values:

```
s.out <- sim(z.out, x = x.out)
summary(s.out)

##
##  sim x :
##  -----
## ev
##       mean     sd  50%  2.5% 97.5%
## [1,] 3.256 0.2405 3.25 2.812 3.728
## pv
##       mean    sd 50% 2.5% 97.5%
## [1,] 3.206 1.878   3    0     7

plot(s.out)
```

2.9.3 Model

Let Y_i be the number of independent events that occur during a fixed time period. This variable can take any non-negative integer.

- The Poisson distribution has *stochastic component*

$$Y_i \ \sim \ \text{Poisson}(\lambda_i),$$

  where λ_i is the mean and variance parameter.

- The *systematic component* is

$$\lambda_i \ = \ \exp(x_i \beta),$$

  where x_i is the vector of explanatory variables, and β is the vector of coefficients.

2.9.4 Quantities of Interest

- The expected value (qi$ev) is the mean of simulations from the stochastic component,

$$E(Y) = \lambda_i = \exp(x_i \beta),$$

  given draws of β from its sampling distribution.

- The predicted value (qi$pr) is a random draw from the poisson distribution defined by mean λ_i.
- The first difference in the expected values (qi$fd) is given by:

$$\text{FD} \ = \ E(Y|x_1) - E(Y \mid x)$$

- In conditional prediction models, the average expected treatment effect (att.ev) for the treatment group is

$$\frac{1}{\sum_{i=1}^n t_i} \sum_{i:t_i=1}^n \left\{ Y_i(t_i = 1) - E[Y_i(t_i = 0)] \right\},$$

  where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. Variation in the simulations are due to uncertainty in simulating $E[Y_i(t_i = 0)]$, the counterfactual expected value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.

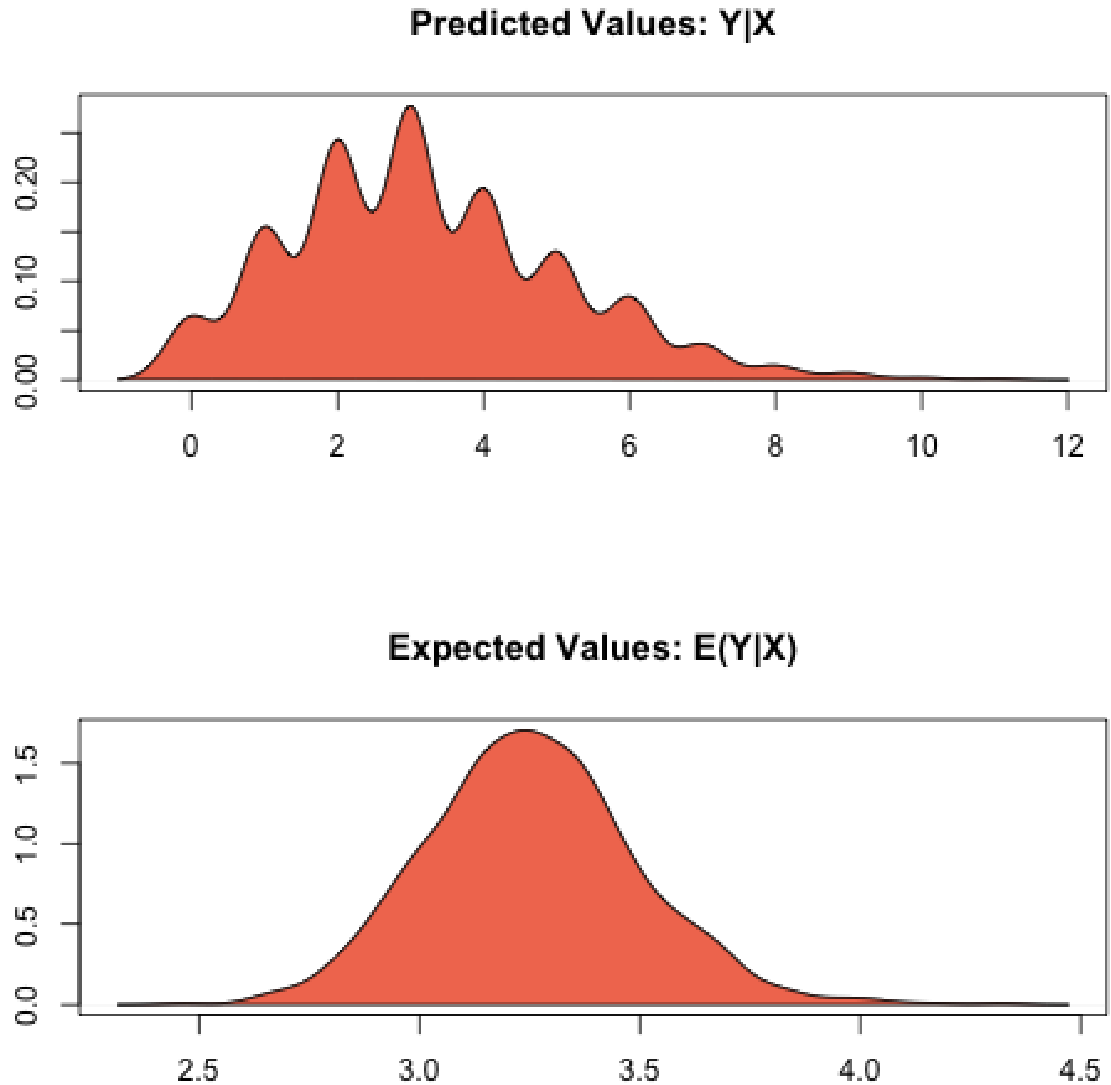


Figure 2.8: Zelig-poisson

- In conditional prediction models, the average predicted treatment effect (att.pr) for the treatment group is

$$\frac{1}{\sum_{i=1}^{n} t_i} \sum_{i:t_i=1}^{n} \left\{ Y_i(t_i = 1) - Y_i(\widehat{t_i = 0}) \right\},$$

  where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. Variation in the simulations are due to uncertainty in simulating $Y_i(\widehat{t_i = 0})$, the counterfactual predicted value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.

2.9.5 Output Values

The output of each Zelig command contains useful information which you may view. For example, if you run `z.out <- zelig(y ~ x, model = poisson, data)`, then you may examine the available information in `z.out` by using `names(z.out)`, see the coefficients by using z.out$coefficients, and a default summary of information through `summary(z.out)`.

2.9.6 See also

The poisson model is part of the stats package by . Advanced users may wish to refer to `help(glm)` and `help(family)`.

2.10 zelig-probit

Probit Regression for Dichotomous Dependent Variables

Use probit regression to model binary dependent variables specified as a function of a set of explanatory variables.

2.10.1 Syntax

With reference classes:

```
z5 <- zprobit$new()
z5$zelig(Y ~ X1 + X ~ X, data = mydata)
z5$setx()
z5$sim()
```

With the Zelig 4 compatibility wrappers:

```
z.out <- zelig(Y ~ X1 + X2, model = "probit", data = mydata)
x.out <- setx(z.out)
s.out <- sim(z.out, x = x.out, x1 = NULL)
```

2.10.2 Example

Attach the sample turnout dataset:

```
data(turnout)
```

Estimate parameter values for the probit regression:

```
z.out <- zelig(vote ~ race + educate, model = "probit", data = turnout)

## How to cite this model in Zelig:
##   Kosuke Imai, Gary King, Olivia Lau. 2007.
##   probit: Probit Regression for Dichotomous Dependent Variables
##   in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
##   http://datascience.iq.harvard.edu/zelig
```

```
summary(z.out)

## Model: 1
## Call:  stats::glm(formula = vote ~ race + educate, family = binomial("probit"),
##     data = .)
##
## Coefficients:
## (Intercept)    racewhite      educate
##     -0.7259       0.2991       0.0971
##
## Degrees of Freedom: 1999 Total (i.e. Null);  1997 Residual
## Null Deviance:       2270
## Residual Deviance: 2140  AIC: 2140
## Next step: Use 'setx' method
```

Set values for the explanatory variables to their default values.

```
x.out <- setx(z.out)
```

Simulate quantities of interest from the posterior distribution.

```
s.out <- sim(z.out, x = x.out)
```

```
summary(s.out)
```

```
plot(s.out1)
```

2.10.3 Model

Let Y_i be the observed binary dependent variable for observation i which takes the value of either 0 or 1.

- The *stochastic component* is given by

$$Y_i \sim \text{Bernoulli}(\pi_i),$$

  where $\pi_i = \Pr(Y_i = 1)$.

- The *systematic component* is

$$\pi_i = \Phi(x_i \beta)$$

  where $\Phi(\mu)$ is the cumulative distribution function of the Normal distribution with mean 0 and unit variance.

2.10.4 Quantities of Interest

- The expected value (qi$ev) is a simulation of predicted probability of success

$$E(Y) = \pi_i = \Phi(x_i \beta),$$

  given a draw of β from its sampling distribution.

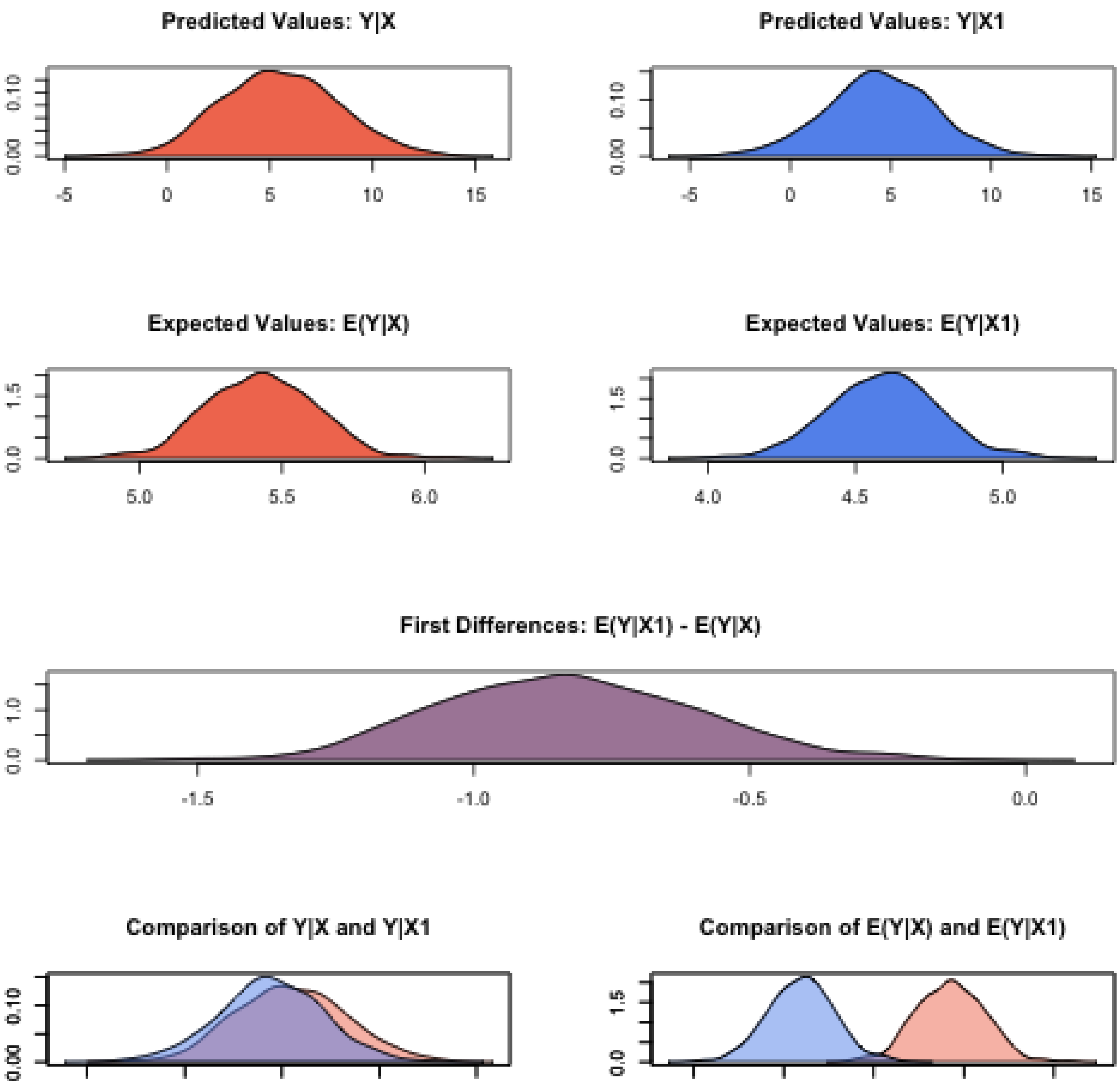


Figure 2.9: Zelig-probit

- The predicted value (qi$pr) is a draw from a Bernoulli distribution with mean π_i.
- The first difference (qi$fd) in expected values is defined as

$$\text{FD} = \Pr(Y = 1 \mid x_1) - \Pr(Y = 1 \mid x).$$

- The risk ratio (qi$rr) is defined as

$$\text{RR} = \Pr(Y = 1 \mid x_1) / \Pr(Y = 1 \mid x).$$

- In conditional prediction models, the average expected treatment effect (att.ev) for the treatment group is

$$\frac{1}{\sum_{i=1}^n t_i} \sum_{i:t_i=1}^n \{Y_i(t_i = 1) - E[Y_i(t_i = 0)]\},$$

  where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. Variation in the simulations are due to uncertainty in simulating $E[Y_i(t_i = 0)]$, the counterfactual expected value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.
- In conditional prediction models, the average predicted treatment effect (att.pr) for the treatment group is

$$\frac{1}{\sum_{i=1}^n t_i} \sum_{i:t_i=1}^n \left\{Y_i(t_i = 1) - \widehat{Y_i(t_i = 0)}\right\},$$

  where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. Variation in the simulations are due to uncertainty in simulating $\widehat{Y_i(t_i = 0)}$, the counterfactual predicted value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.

2.10.5 Output Values

The output of each Zelig command contains useful information which you may view. For example, if you run `z.out <- zelig(y ~ x, model = probit, data)`, then you may examine the available information in `z.out` by using `names(z.out)`, see the coefficients by using z.out$coefficients, and a default summary of information through `summary(z.out)`.

2.10.6 See also

The probit model is part of the stats package by . Advanced users may wish to refer to `help(glm)` and `help(family)`.

2.11 zelig-relogit

Rare Events Logistic Regression for Dichotomous Dependent Variables

The relogit procedure estimates the same model as standard logistic regression (appropriate when you have a dichotomous dependent variable and a set of explanatory variables; see), but the estimates are corrected for the bias that occurs when the sample is small or the observed events are rare (i.e., if the dependent variable has many more 1s than 0s or the reverse). The relogit procedure also optionally uses prior correction for case-control sampling designs.

2.11.1 Syntax

With reference classes:

```
z5 <- zrelogit$new()
z5$zelig(Y ~ X1 + X2, tau = NULL,
                      case.control = c("prior", "weighting"),
                      bias.correct = TRUE, robust = FALSE,
                      data = mydata, ...)
z5$setx()
z5$sim()
```

With the Zelig 4 compatibility wrappers:

```
z.out <- zelig(Y ~ X1 + X2, model = "relogit", tau = NULL,
                      case.control = c("prior", "weighting"),
                      bias.correct = TRUE, robust = FALSE,
                      data = mydata, ...)
x.out <- setx(z.out)
s.out <- sim(z.out, x = x.out)
```

2.11.2 Arguments

The relogit procedure supports four optional arguments in addition to the standard arguments for zelig(). You may additionally use:

- tau: a vector containing either one or two values for τ, the true population fraction of ones. Use, for example, tau = c(0.05, 0.1) to specify that the lower bound on tau is 0.05 and the upper bound is 0.1. If left unspecified, only finite-sample bias correction is performed, not case-control correction.
- case.control: if tau is specified, choose a method to correct for case-control sampling design: “prior” (default) or “weighting”.
- bias.correct: a logical value of TRUE (default) or FALSE indicating whether the intercept should be corrected for finite sample (rare events) bias.

Note that if tau = NULL, bias.correct = FALSE, the relogit procedure performs a standard logistic regression without any correction.

2.11.3 Example 1: One Tau with Prior Correction and Bias Correction

Due to memory and space considerations, the data used here are a sample drawn from the full data set used in King and Zeng, 2001, The proportion of militarized interstate conflicts to the absence of disputes is $\tau = 1,042/303,772 \approx 0.00343$. To estimate the model,

```
data(mid)
```

```
z.out1 <- zelig(conflict ~ major + contig + power + maxdem + mindem + years, data = mid, model = "re
```

```
## How to cite this model in Zelig:
##   Kosuke Imai, Gary King, and Olivia Lau. 2014.
##   relogit: Rare Events Logistic Regression for Dichotomous Dependent Variables
##   in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
##   http://datascience.iq.harvard.edu/zelig
```

Summarize the model output:

```
summary(z.out1)

## Model: 1
## Call:  relogit(formula = cbind(conflict, 1 - conflict) ~ major + contig +
##     power + maxdem + mindem + years, data = ., tau = 0.00343020423212146,
##     bias.correct = TRUE, case.control = "prior")
##
## Coefficients:
## (Intercept)        major       contig        power       maxdem
##     -7.5084       2.4320       4.1080       1.0536       0.0480
##      mindem        years
##     -0.0641      -0.0629
##
## Degrees of Freedom: 3125 Total (i.e. Null);  3119 Residual
## Null Deviance:       3980
## Residual Deviance: 1870  AIC: 1880
## Next step: Use 'setx' method
```

Set the explanatory variables to their means:

```
x.out1 <- setx(z.out1)
```

Simulate quantities of interest:

```
s.out1 <- sim(z.out1, x = x.out1)
summary(s.out1)

##
##  sim x :
##  -----
## ev
##          mean       sd      50%     2.5%    97.5%
## [1,] 0.002393 0.000155 0.002388 0.002103 0.002708
## pv
##          0     1
## [1,] 0.998 0.002

plot(s.out1)
```

2.11.4 Example 2: One Tau with Weighting, Robust Standard Errors, and Bias Correction

Suppose that we wish to perform case control correction using weighting (rather than the default prior correction). To estimate the model:

```
z.out2 <- zelig(conflict ~ major + contig + power + maxdem + mindem + years, data = mid, model = "rel

## Error: unused argument (robust = TRUE)
```

Summarize the model output:

```
summary(z.out2)

## Model: 1
## Call:
## stats::lm(formula = unem ~ gdp + trade + capmob + as.factor(country),
##     data = .)
```

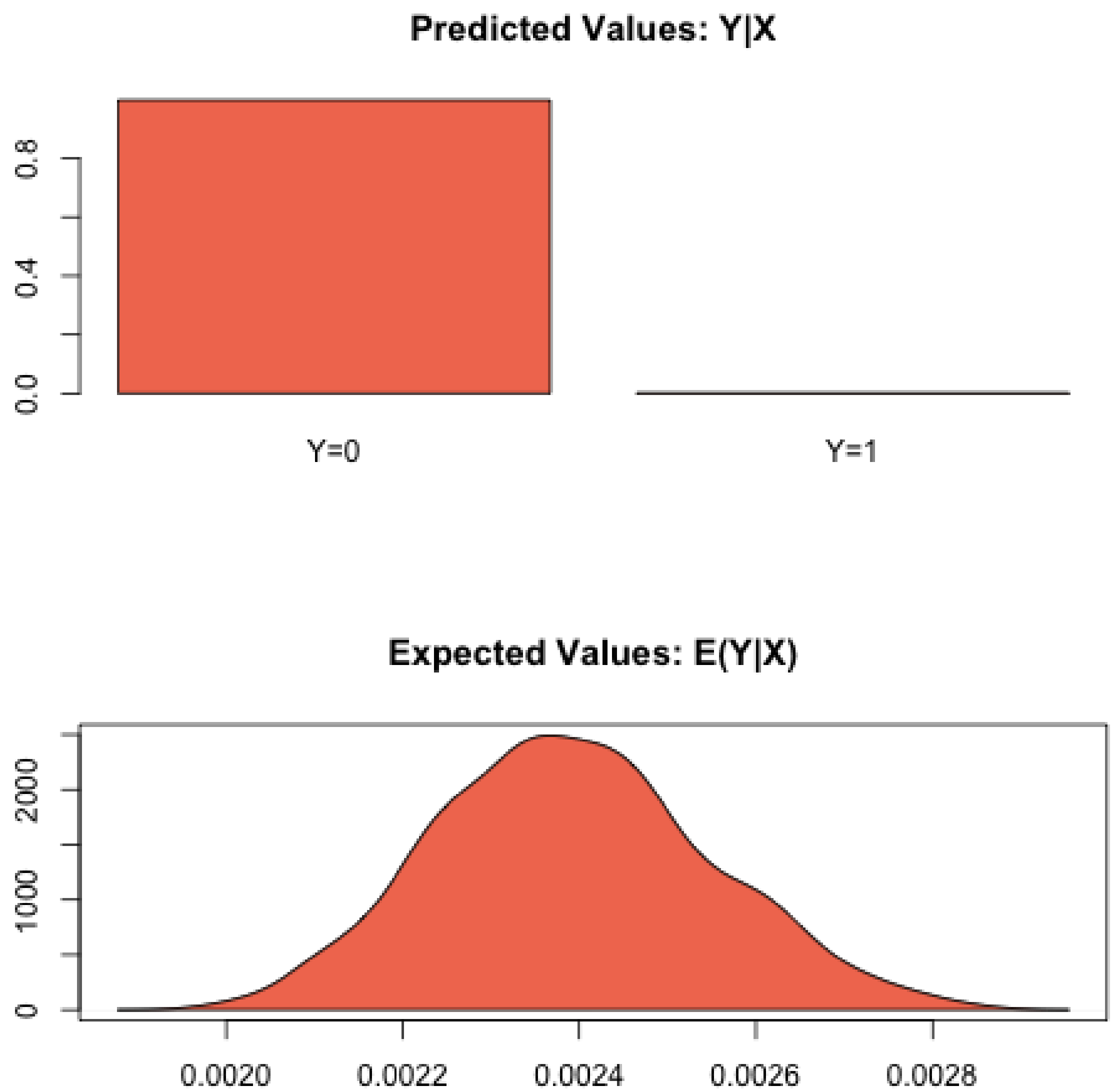


Figure 2.10: Zelig-relogit

```
##
## Coefficients:
##                    (Intercept)                             gdp
##                         -5.843                          -0.110
##                          trade                          capmob
##                          0.144                           0.815
##       as.factor(country)Belgium        as.factor(country)Canada
##                         -1.599                           6.759
##       as.factor(country)Denmark       as.factor(country)Finland
##                          4.311                           4.810
##        as.factor(country)France         as.factor(country)Italy
##                          6.905                           9.290
##         as.factor(country)Japan   as.factor(country)Netherlands
##                          5.459                          -1.459
##        as.factor(country)Norway        as.factor(country)Sweden
##                         -2.754                           0.925
## as.factor(country)United Kingdom   as.factor(country)United States
##                          5.601                          10.066
##   as.factor(country)West Germany
##                          3.364
##
## Next step: Use 'setx' method
```

Set the explanatory variables to their means:

```
x.out2 <- setx(z.out2)
```

Simulate quantities of interest:

```
s.out2 <- sim(z.out2, x = x.out2)
summary(s.out2)
```

```
##
##  sim x :
##  -----
## ev
##    mean     sd  50%  2.5% 97.5%
## 1 10.59 0.4011 10.6 9.818  11.4
## pv
##    mean     sd  50%  2.5% 97.5%
## 1 10.59 0.4011 10.6 9.818  11.4
```

2.11.5 Example 3: Two Taus with Bias Correction and Prior Correction

Suppose that we did not know that $\tau \approx 0.00343$, but only that it was somewhere between $(0.002, 0.005)$. To estimate a model with a range of feasible estimates for τ (using the default prior correction method for case control correction):

```
z.out2 <- zelig(conflict ~ major + contig + power + maxdem + mindem + years, data = mid, model = "rel
```

```
## How to cite this model in Zelig:
##   Kosuke Imai, Gary King, and Olivia Lau. 2014.
##   relogit: Rare Events Logistic Regression for Dichotomous Dependent Variables
##   in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
##   http://datascience.iq.harvard.edu/zelig
```

Summarize the model output:

```
z.out2
```

```
## Model: 1$lower.estimate
##
## Call:  (function (formula, data = sys.parent(), tau = NULL, bias.correct = TRUE,
##     case.control = "prior", ...)
## {
##     mf <- match.call()
##     mf$tau <- mf$bias.correct <- mf$case.control <- NULL
##     if (!is.null(tau)) {
##         tau <- unique(tau)
##         if (length(case.control) > 1)
##             stop("You can only choose one option for case control correction.")
##         ck1 <- grep("p", case.control)
##         ck2 <- grep("w", case.control)
##         if (length(ck1) == 0 & length(ck2) == 0)
##             stop("choose either case.control = \"prior\" ", "or case.control = \"weighting\"")
##         if (length(ck2) == 0)
##             weighting <- FALSE
##         else weighting <- TRUE
##     }
##     else weighting <- FALSE
##     if (length(tau) > 2)
##         stop("tau must be a vector of length less than or equal to 2")
##     else if (length(tau) == 2) {
##         mf[[1]] <- relogit
##         res <- list()
##         mf$tau <- min(tau)
##         res$lower.estimate <- eval(as.call(mf), parent.frame())
##         mf$tau <- max(tau)
##         res$upper.estimate <- eval(as.call(mf), parent.frame())
##         res$formula <- formula
##         class(res) <- c("Relogit2", "Relogit")
##         return(res)
##     }
##     else {
##         mf[[1]] <- glm
##         mf$family <- binomial(link = "logit")
##         y2 <- model.response(model.frame(mf$formula, data))
##         if (is.matrix(y2))
##             y <- y2[, 1]
##         else y <- y2
##         ybar <- mean(y)
##         if (weighting) {
##             w1 <- tau/ybar
##             w0 <- (1 - tau)/(1 - ybar)
##             wi <- w1 * y + w0 * (1 - y)
##             mf$weights <- wi
##         }
##         res <- eval(as.call(mf), parent.frame())
##         res$call <- match.call(expand.dots = TRUE)
##         res$tau <- tau
##         X <- model.matrix(res)
##         if (bias.correct) {
##             pihat <- fitted(res)
##             if (is.null(tau))
##                 wi <- rep(1, length(y))
##             else if (weighting)
```

```
##                 res$weighting <- TRUE
##             else {
##                 w1 <- tau/ybar
##                 w0 <- (1 - tau)/(1 - ybar)
##                 wi <- w1 * y + w0 * (1 - y)
##                 res$weighting <- FALSE
##             }
##             W <- pihat * (1 - pihat) * wi
##             Qdiag <- lm.influence(lm(y ~ X - 1, weights = W))$hat/W
##             if (is.null(tau))
##                 xi <- 0.5 * Qdiag * (2 * pihat - 1)
##             else xi <- 0.5 * Qdiag * ((1 + w0) * pihat - w0)
##             res$coefficients <- res$coefficients - lm(xi ~ X -
##                 1, weights = W)$coefficients
##             res$bias.correct <- TRUE
##         }
##         else res$bias.correct <- FALSE
##         if (!is.null(tau) & !weighting) {
##             if (tau <= 0 || tau >= 1)
##                 stop("\ntau needs to be between 0 and 1.\n")
##             res$coefficients["(Intercept)"] <- res$coefficients["(Intercept)"] -
##                 log(((1 - tau)/tau) * (ybar/(1 - ybar)))
##             res$prior.correct <- TRUE
##             res$weighting <- FALSE
##         }
##         else res$prior.correct <- FALSE
##         if (is.null(res$weighting))
##             res$weighting <- FALSE
##         res$linear.predictors <- t(res$coefficients) %*% t(X)
##         res$fitted.values <- 1/(1 + exp(-res$linear.predictors))
##         res$zelig <- "Relogit"
##         class(res) <- c("Relogit", "glm")
##         return(res)
##     }
## })(formula = cbind(conflict, 1 - conflict) ~ major + contig +
##     power + maxdem + mindem + years, data = ., tau = 0.002)
##
## Coefficients:
## (Intercept)        major       contig        power       maxdem
##     -8.0492       2.4320       4.1079       1.0536       0.0480
##      mindem        years
##     -0.0641      -0.0629
##
## Degrees of Freedom: 3125 Total (i.e. Null);  3119 Residual
## Null Deviance:       3980
## Residual Deviance: 1870  AIC: 1880
##
## $upper.estimate
##
## Call:  (function (formula, data = sys.parent(), tau = NULL, bias.correct = TRUE,
##     case.control = "prior", ...)
## {
##     mf <- match.call()
##     mf$tau <- mf$bias.correct <- mf$case.control <- NULL
##     if (!is.null(tau)) {
##         tau <- unique(tau)
##         if (length(case.control) > 1)
##             stop("You can only choose one option for case control correction.")
```

```
##         ck1 <- grep("p", case.control)
##         ck2 <- grep("w", case.control)
##         if (length(ck1) == 0 & length(ck2) == 0)
##             stop("choose either case.control = \"prior\" ", "or case.control = \"weighting\"")
##         if (length(ck2) == 0)
##             weighting <- FALSE
##         else weighting <- TRUE
##     }
##     else weighting <- FALSE
##     if (length(tau) > 2)
##         stop("tau must be a vector of length less than or equal to 2")
##     else if (length(tau) == 2) {
##         mf[[1]] <- relogit
##         res <- list()
##         mf$tau <- min(tau)
##         res$lower.estimate <- eval(as.call(mf), parent.frame())
##         mf$tau <- max(tau)
##         res$upper.estimate <- eval(as.call(mf), parent.frame())
##         res$formula <- formula
##         class(res) <- c("Relogit2", "Relogit")
##         return(res)
##     }
##     else {
##         mf[[1]] <- glm
##         mf$family <- binomial(link = "logit")
##         y2 <- model.response(model.frame(mf$formula, data))
##         if (is.matrix(y2))
##             y <- y2[, 1]
##         else y <- y2
##         ybar <- mean(y)
##         if (weighting) {
##             w1 <- tau/ybar
##             w0 <- (1 - tau)/(1 - ybar)
##             wi <- w1 * y + w0 * (1 - y)
##             mf$weights <- wi
##         }
##         res <- eval(as.call(mf), parent.frame())
##         res$call <- match.call(expand.dots = TRUE)
##         res$tau <- tau
##         X <- model.matrix(res)
##         if (bias.correct) {
##             pihat <- fitted(res)
##             if (is.null(tau))
##                 wi <- rep(1, length(y))
##             else if (weighting)
##                 res$weighting <- TRUE
##             else {
##                 w1 <- tau/ybar
##                 w0 <- (1 - tau)/(1 - ybar)
##                 wi <- w1 * y + w0 * (1 - y)
##                 res$weighting <- FALSE
##             }
##             W <- pihat * (1 - pihat) * wi
##             Qdiag <- lm.influence(lm(y ~ X - 1, weights = W))$hat/W
##             if (is.null(tau))
##                 xi <- 0.5 * Qdiag * (2 * pihat - 1)
##             else xi <- 0.5 * Qdiag * ((1 + w0) * pihat - w0)
##             res$coefficients <- res$coefficients - lm(xi ~ X -
```

```
##                  1, weights = W)$coefficients
##              res$bias.correct <- TRUE
##          }
##          else res$bias.correct <- FALSE
##          if (!is.null(tau) & !weighting) {
##              if (tau <= 0 || tau >= 1)
##                  stop("\ntau needs to be between 0 and 1.\n")
##              res$coefficients["(Intercept)"] <- res$coefficients["(Intercept)"] -
##                  log(((1 - tau)/tau) * (ybar/(1 - ybar)))
##              res$prior.correct <- TRUE
##              res$weighting <- FALSE
##          }
##          else res$prior.correct <- FALSE
##          if (is.null(res$weighting))
##              res$weighting <- FALSE
##          res$linear.predictors <- t(res$coefficients) %*% t(X)
##          res$fitted.values <- 1/(1 + exp(-res$linear.predictors))
##          res$zelig <- "Relogit"
##          class(res) <- c("Relogit", "glm")
##          return(res)
##      }
## })(formula = cbind(conflict, 1 - conflict) ~ major + contig +
##     power + maxdem + mindem + years, data = ., tau = 0.005)
##
## Coefficients:
## (Intercept)        major       contig        power       maxdem
##     -7.1300       2.4320       4.1080       1.0536       0.0480
##      mindem        years
##     -0.0641      -0.0629
##
## Degrees of Freedom: 3125 Total (i.e. Null);  3119 Residual
## Null Deviance:       3980
## Residual Deviance: 1870  AIC: 1880
##
## $formula
## cbind(conflict, 1 - conflict) ~ major + contig + power + maxdem +
##     mindem + years
## <environment: 0x7f8eccf6a9c8>
##
## attr(,"class")
## [1] "Relogit2" "Relogit"
## Next step: Use 'setx' method
```

Set the explanatory variables to their means:

```
x.out2 <- setx(z.out2)
```

Simulate quantities of interest:

```
s.out <- sim(z.out2, x = x.out2)
```

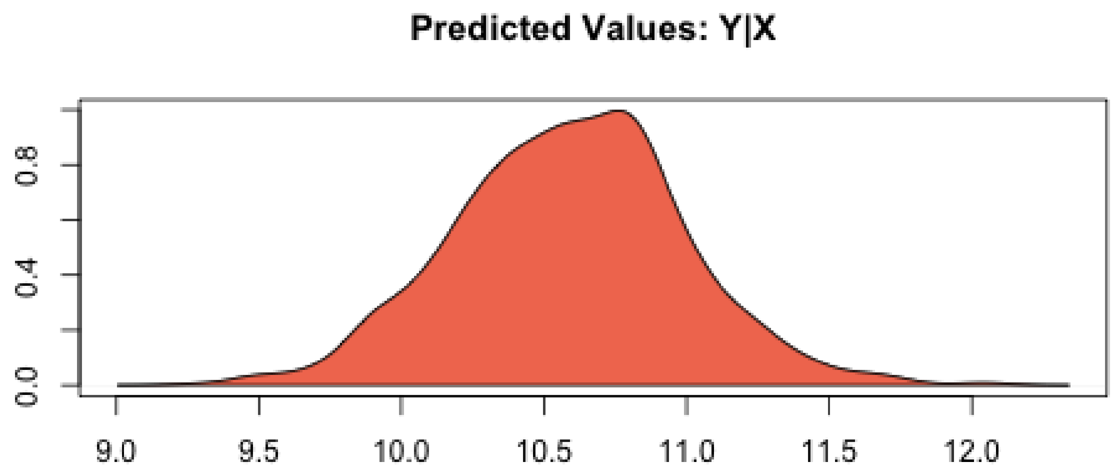
```
## Error: no applicable method for 'vcov' applied to an object of class
## "c('Relogit2', 'Relogit')"
```

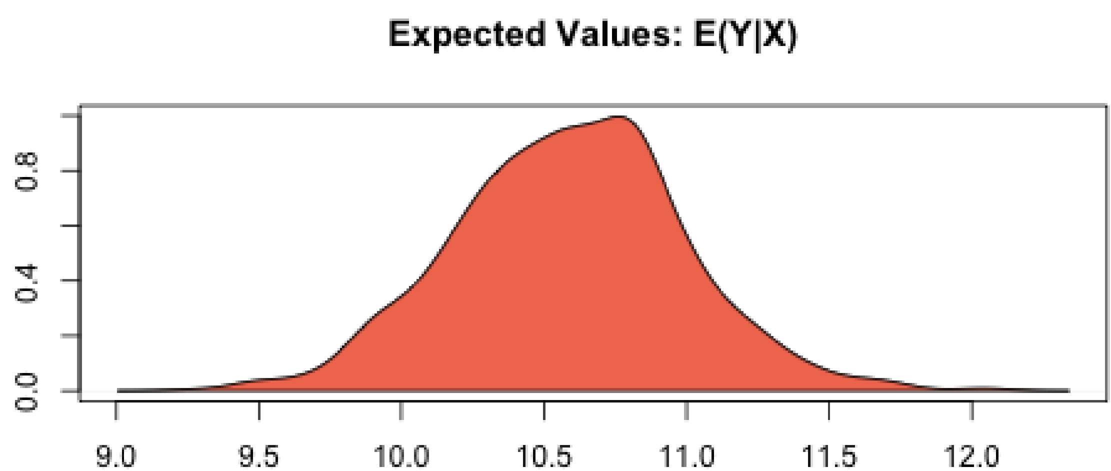
```
summary(s.out2)
```

```
##
##  sim x :
##  -----
```

```
## ev
##    mean     sd  50%  2.5% 97.5%
## 1 10.59 0.4011 10.6 9.818  11.4
## pv
##    mean     sd  50%  2.5% 97.5%
## 1 10.59 0.4011 10.6 9.818  11.4
```

```
plot(s.out2)
```





The cost of giving a range of values for τ is that point estimates are not available for quantities of interest. Instead, quantities are presented as confidence intervals with significance less than or equal to a specified level (e.g., at least 95% of the simulations are contained in the nominal 95% confidence interval).

2.11.6 Model

- Like the standard logistic regression, the *stochastic component* for the rare events logistic regression is:

$$Y_i \sim \text{Bernoulli}(\pi_i),$$

where Y_i is the binary dependent variable, and takes a value of either 0 or 1.

- The *systematic component* is:

$$\pi_i = \frac{1}{1 + \exp(-x_i \beta)}.$$

- If the sample is generated via a case-control (or choice-based) design, such as when drawing all events (or "cases") and a sample from the non-events (or "controls") and going backwards to collect the explanatory variables, you must correct for selecting on the dependent variable. While the slope coefficients are approximately unbiased, the constant term may be significantly biased. Zelig has two methods for case control correction:

  1. The "prior correction" method adjusts the intercept term. Let τ be the true population fraction of events, $\bar{y}$ the fraction of events in the sample, and $\hat{\beta_0}$ the uncorrected intercept term. The corrected intercept β_0 is:

  $$\beta = \hat{\beta_0} - \ln \left[\left(\frac{1-\tau}{\tau} \right) \left(\frac{\bar{y}}{1-\bar{y}} \right) \right].$$

  2. The "weighting" method performs a weighted logistic regression to correct for a case-control sampling design. Let the 1 subscript denote observations for which the dependent variable is observed as a 1, and the 0 subscript denote observations for which the dependent variable is observed as a 0. Then the vector of weights w_i

  $$\begin{aligned} w_1 &= \frac{\tau}{\bar{y}} \\ w_0 &= \frac{(1-\tau)}{(1-\bar{y})} \\ w_i &= w_1 Y_i + w_0 (1 - Y_i) \end{aligned}$$

  If τ is unknown, you may alternatively specify an upper and lower bound for the possible range of τ. In this case, the relogit procedure uses "robust Bayesian" methods to generate a confidence interval (rather than a point estimate) for each quantity of interest. The nominal coverage of the confidence interval is at least as great as the actual coverage.

- By default, estimates of the the coefficients β are bias-corrected to account for finite sample or rare events bias. In addition, quantities of interest, such as predicted probabilities, are also corrected of rare-events bias. If $\widehat{\beta}$ are the uncorrected logit coefficients and bias($\widehat{\beta}$) is the bias term, the corrected coefficients $\tilde{\beta}$ are

  $$\widehat{\beta} - \text{bias}(\widehat{\beta}) = \tilde{\beta}$$

  The bias term is

  $$\text{bias}(\widehat{\beta}) = (X'WX)^{-1} X'W\xi$$

  where

  $$\begin{aligned} \xi_i &= 0.5 Q_{ii} \Big((1 + w - 1)\widehat{\pi}_i - w_1 \Big) \\ Q &= X(X'WX)^{-1}X' \\ W &= \text{diag}\{\widehat{\pi}_i (1 - \widehat{\pi}_i) w_i\} \end{aligned}$$

  where w_i and w_1 are given in the "weighting" section above.

2.11.7 Quantities of Interest

- For either one or no τ:
  - The expected values (qi$ev) for the rare events logit are simulations of the predicted probability

    $$E(Y) = \pi_i = \frac{1}{1 + \exp(-x_i \beta)},$$

    given draws of β from its posterior.
  - The predicted value (qi$pr) is a draw from a binomial distribution with mean equal to the simulated π_i.
  - The first difference (qi$fd) is defined as

    $$\text{FD} = \Pr(Y = 1 \mid x_1, \tau) - \Pr(Y = 1 \mid x, \tau).$$

  - The risk ratio (qi$rr) is defined as

    $$\text{RR} = \Pr(Y = 1 \mid x_1, \tau) \ / \ \Pr(Y = 1 \mid x, \tau).$$

- For a range of τ defined by $[\tau_1, \tau_2]$, each of the quantities of interest are $n \times 2$ matrices, which report the lower and upper bounds, respectively, for a confidence interval with nominal coverage at least as great as the actual coverage. At worst, these bounds are conservative estimates for the likely range for each quantity of interest. Please refer to for the specific method of calculating bounded quantities of interest.
- In conditional prediction models, the average expected treatment effect (att.ev) for the treatment group is

  $$\frac{1}{\sum_{i=1}^n t_i} \sum_{i:t_i=1}^n \left\{ Y_i(t_i = 1) - E[Y_i(t_i = 0)] \right\},$$

  where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. Variation in the simulations are due to uncertainty in simulating $E[Y_i(t_i = 0)]$, the counterfactual expected value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.
- In conditional prediction models, the average predicted treatment effect (att.pr) for the treatment group is

  $$\frac{1}{\sum_{i=1}^n t_i} \sum_{i:t_i=1}^n \left\{ Y_i(t_i = 1) - \widehat{Y_i(t_i = 0)} \right\},$$

  where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. Variation in the simulations are due to uncertainty in simulating $\widehat{Y_i(t_i = 0)}$, the counterfactual predicted value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.

2.11.8 Output Values

The output of each Zelig command contains useful information which you may view. For example, if you run `z.out <- zelig(y ~ x, model = relogit, data)`, then you may examine the available information in `z.out` by using `names(z.out)`, see the coefficients by using z.out$coefficients, and a default summary of information through `summary(z.out)`.

2.11.9 Differences with Stata Version

The Stata version of ReLogit and the R implementation differ slightly in their coefficient estimates due to differences in the matrix inversion routines implemented in R and Stata. Zelig uses orthogonal-triangular decomposition (through lm.influence()) to compute the bias term, which is more numerically stable than standard matrix calculations.

2.11.10 See also

2.12 zelig-tobit

Linear Regression for a Left-Censored Dependent Variable

Tobit regression estimates a linear regression model for a left-censored dependent variable, where the dependent variable is censored from below. While the classical tobit model has values censored at 0, you may select another censoring point. For other linear regression models with fully observed dependent variables, see Bayesian regression (), maximum likelihood normal regression (), or least squares ().

2.12.1 Syntax

```
z5 <- ztobit$new()
z5$zelig(Y ~ X1 + X2, below = 0, above = Inf, data = mydata)
z5$setx()
z5$sim()
```

With the Zelig 4 compatibility wrappers:

```
z.out <- zelig(Y ~ X1 + X2, below = 0, above = Inf, model = "tobit", data = mydata)
x.out <- setx(z.out)
s.out <- sim(z.out, x = x.out)
```

2.12.2 Inputs

zelig() accepts the following arguments to specify how the dependent variable is censored.

- `below`: (defaults to 0) The point at which the dependent variable is censored from below. If any values in the dependent variable are observed to be less than the censoring point, it is assumed that that particular observation is censored from below at the observed value. (See for a Bayesian implementation that supports both left and right censoring.)
- robust: defaults to FALSE. If TRUE, zelig() computes robust standard errors based on sandwich estimators (see and) and the options selected in cluster.
- cluster: if robust = TRUE, you may select a variable to define groups of correlated observations. Let x3 be a variable that consists of either discrete numeric values, character strings, or factors that define strata. Then

  ```
  > z.out <- zelig(y ~ x1 + x2, robust = TRUE, cluster = "x3",
                   model = "tobit", data = mydata)
  ```

  means that the observations can be correlated within the strata defined by the variable x3, and that robust standard errors should be calculated according to those clusters. If robust = TRUE but cluster is not specified, zelig() assumes that each observation falls into its own cluster.

Zelig users may wish to refer to `help(survreg)` for more information.

2.12.3 Examples

Basic Example

Attaching the sample dataset:

```
data(tobin)
```

Estimating linear regression using `tobit`:

```
z.out <- zelig(durable ~ age + quant, model = "tobit", data = tobin)
```

```
## How to cite this model in Zelig:
##   Kosuke Imai, Gary King, Olivia Lau. 2011.
##   tobit: Linear regression for Left-Censored Dependent Variable
##   in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
##   http://datascience.iq.harvard.edu/zelig
```

Setting values for the explanatory variables to their sample averages:

```
x.out <- setx(z.out)
```

Simulating quantities of interest from the posterior distribution given `x.out`.

```
s.out1 <- sim(z.out, x = x.out)
```

```
summary(s.out1)
```

```
##
##  sim x :
##  -----
## ev
##   mean     sd   50%   2.5% 97.5%
## 1 1.55 0.6187 1.476 0.5527 2.965
## pv
##      mean    sd  50% 2.5% 97.5%
## [1,] 3.23 4.082 1.81    0  13.8
```

Simulating First Differences

Set explanatory variables to their default(mean/mode) values, with high (80th percentile) and low (20th percentile) liquidity ratio (`quant`):

```
x.high <- setx(z.out, quant = quantile(tobin$quant, prob = 0.8))
x.low <- setx(z.out, quant = quantile(tobin$quant, prob = 0.2))
```

Estimating the first difference for the effect of high versus low liquidity ratio on duration(`durable`):

```
s.out2 <- sim(z.out, x = x.high, x1 = x.low)
```

```
summary(s.out2)
```

```
##
##  sim x :
##  -----
## ev
##   mean     sd   50%   2.5% 97.5%
## 1 1.18 0.7514 1.041 0.1329 2.995
## pv
##       mean    sd   50% 2.5% 97.5%
## [1,] 2.884 3.884 1.182    0 13.02
##
##  sim x1 :
##  -----
```

```
## ev
##   mean     sd   50%   2.5% 97.5%
## 1 2.06 0.9851 1.911 0.6058 4.218
## pv
##       mean    sd  50% 2.5% 97.5%
## [1,] 3.495 4.289 1.98    0 14.66
## fd
##     mean    sd    50%   2.5% 97.5%
## 1 0.8797 1.184 0.8411 -1.315 3.409
```

```
plot(s.out1)
```

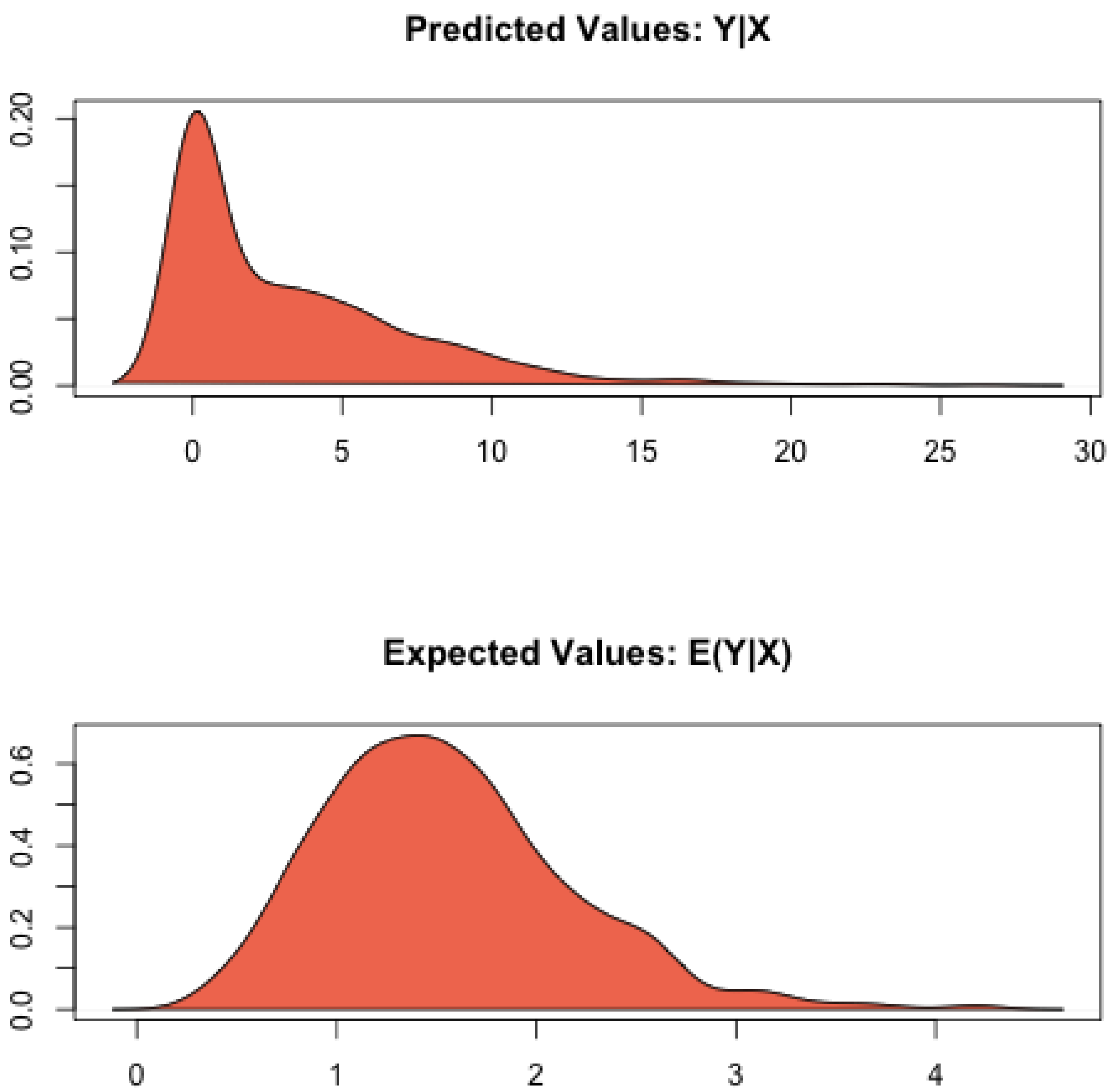


Figure 2.11: Zelig-tobit

2.12.4 Model

- Let Y_i^* be a latent dependent variable which is distributed with *stochastic* component

$$Y_i^* \sim \quad \text{Normal}(\mu_i, \sigma^2)$$

  where μ_i is a vector means and σ^2 is a scalar variance parameter. Y_i^* is not directly observed, however. Rather we observed Y_i which is defined as:

$$Y_i = \begin{cases} Y_i^* & \text{if} \quad c < Y_i^* \\ c & \text{if} \quad c \geq Y_i^* \end{cases}$$

  where c is the lower bound below which Y_i^* is censored.

- The *systematic component* is given by

$$\mu_i = \quad x_i \beta,$$

  where x_i is the vector of k explanatory variables for observation i and β is the vector of coefficients.

2.12.5 Quantities of Interest

- The expected values (`qi$ev`) for the tobit regression model are the same as the expected value of $Y*$:

$$E(Y^*|X) = \mu_i = x_i \beta$$

- The first difference (`qi$fd`) for the tobit regression model is defined as

$$\text{FD} = E(Y^* \mid x_1) - E(Y^* \mid x).$$

- In conditional prediction models, the average expected treatment effect (`qi$att.ev`) for the treatment group is

$$\frac{1}{\sum t_i} \sum_{i:t_i=1} [E[Y_i^*(t_i = 1)] - E[Y_i^*(t_i = 0)]],$$

  where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups.

2.12.6 Output Values

The output of each Zelig command contains useful information which you may view. For example, if you run:

```
z.out <- zelig(y ~ x, model = "tobit", data)
```

then you may examine the available information in ‘‘z.out‘.

2.12.7 See also

The tobit function is part of the survival library by Terry Therneau, ported to R by Thomas Lumley. Advanced users may wish to refer to `help(survfit)` in the survival library.

zelig-exp

Exponential Regression for Duration Dependent Variables

Use the exponential duration regression model if you have a dependent variable representing a duration (time until an event). The model assumes a constant hazard rate for all events. The dependent variable may be censored (for observations have not yet been completed when data were collected).

Syntax

With reference classes:

```
z5 <- zexp$new()
z5$zelig(Surv(Y, C) ~ X, data = mydata)
z5$setx()
z5$sim()
```

With the Zelig 4 compatibility wrappers:

```
z.out <- zelig(Surv(Y, C) ~ X, model = "exp", data = mydata)
x.out <- setx(z.out)
s.out <- sim(z.out, x = x.out)
```

Exponential models require that the dependent variable be in the form Surv(Y, C), where Y and C are vectors of length n. For each observation i in 1, $\ldots$, n, the value y_i is the duration (lifetime, for example), and the associated c_i is a binary variable such that $c_i = 1$ if the duration is not censored (*e.g.*, the subject dies during the study) or $c_i = 0$ if the duration is censored (*e.g.*, the subject is still alive at the end of the study and is know to live at least as long as y_i). If c_i is omitted, all Y are assumed to be completed; that is, time defaults to 1 for all observations.

Input Values

In addition to the standard inputs, zelig() takes the following additional options for exponential regression:

- robust: defaults to FALSE. If TRUE, zelig() computes robust standard errors based on sandwich estimators (see and) and the options selected in cluster.
- cluster: if robust = TRUE, you may select a variable to define groups of correlated observations. Let x3 be a variable that consists of either discrete numeric values, character strings, or factors that define strata. Then

```
z.out <- zelig(y ~ x1 + x2, robust = TRUE, cluster = "x3",
                         model = "exp", data = mydata)
```

means that the observations can be correlated within the strata defined by the variable x3, and that robust standard errors should be calculated according to those clusters. If robust = TRUE but cluster is not specified, zelig() assumes that each observation falls into its own cluster.

Example

Attach the sample data:

```
data(coalition)
```

Estimate the model:

```
z.out <- zelig(Surv(duration, ciep12) ~ fract + numst2, model = "exp", data = coalition)
```

```
## How to cite this model in Zelig:
##   Olivia Lau, Kosuke Imai, Gary King. 2011.
##   exp: Exponential Regression for Duration Dependent Variables
##   in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
##   http://datascience.iq.harvard.edu/zelig
```

View the regression output:

```
summary(z.out)

## Model: 1Call:
## survival::survreg(formula = Surv(duration, ciep12) ~ fract +
##     numst2, data = ., dist = "exponential", model = FALSE)
##
## Coefficients:
## (Intercept)       fract      numst2
##    5.535873   -0.003909    0.461179
##
## Scale fixed at 1
##
## Loglik(model)= -1077   Loglik(intercept only)= -1101
##  Chisq= 46.66 on 2 degrees of freedom, p= 7.4e-11
## n= 314
## Next step: Use 'setx' method
```

Set the baseline values (with the ruling coalition in the minority) and the alternative values (with the ruling coalition in the majority) for X:

```
x.low <- setx(z.out, numst2 = 0)
x.high <- setx(z.out, numst2 = 1)
```

Simulate expected values and first differences:

```
s.out <- sim(z.out, x = x.low, x1 = x.high)
```

Summarize quantities of interest and produce some plots:

```
summary(s.out)

##
##  sim x :
##  -----
## ev
##    mean    sd   50%  2.5% 97.5%
## 1 15.35 1.529 15.29 12.71 18.36
## pv
##       mean   sd   50%   2.5% 97.5%
## [1,] 15.49 15.8 10.57 0.3662 53.73
##
##  sim x1 :
##  -----
## ev
##    mean    sd  50%  2.5% 97.5%
## 1 24.35 2.029 24.3 20.65 28.56
## pv
##       mean    sd   50%   2.5% 97.5%
## [1,] 25.65 25.85 16.46 0.4663 95.32
## fd
##    mean    sd  50%  2.5% 97.5%
```

```
## 1 9.001 2.515 9.06 3.816 13.91
```

```
plot(s.out)
```

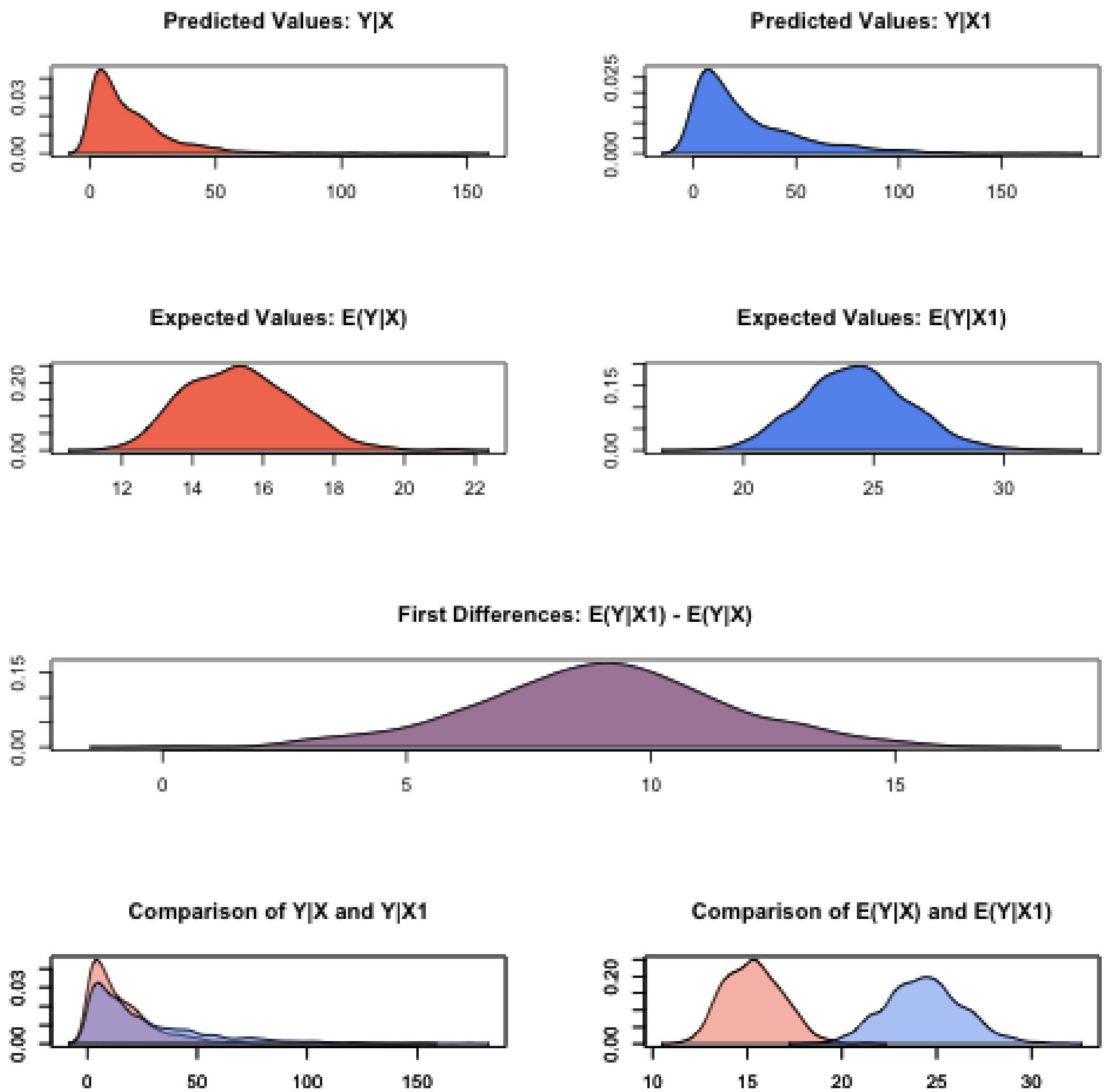


Figure 2.12: Zelig-exp

Model

Let Y_i^* be the survival time for observation i. This variable might be censored for some observations at a fixed time y_c such that the fully observed dependent variable, Y_i, is defined as

$$Y_i = \begin{cases} Y_i^* & \text{if } Y_i^* \leq y_c \\ y_c & \text{if } Y_i^* > y_c \end{cases}$$

- The *stochastic component* is described by the distribution of the partially observed variable Y^*. We assume Y_i^* follows the exponential distribution whose density function is given by

$$f(y_i^* \mid \lambda_i) = \frac{1}{\lambda_i} \exp\left(-\frac{y_i^*}{\lambda_i}\right)$$

for $y_i^* \geq 0$ and $\lambda_i > 0$. The mean of this distribution is λ_i.

In addition, survival models like the exponential have three additional properties. The hazard function $h(t)$ measures the probability of not surviving past time t given survival up to t. In general, the hazard function is equal to $f(t)/S(t)$ where the survival function $S(t) = 1 - \int_0^t f(s)ds$ represents the fraction still surviving at time t. The cumulative hazard function $H(t)$ describes the probability of dying before time t. In general, $H(t) = \int_0^t h(s)ds = -\log S(t)$. In the case of the exponential model,

$$\begin{aligned} h(t) &= \frac{1}{\lambda_i} \\ S(t) &= \exp\left(-\frac{t}{\lambda_i}\right) \\ H(t) &= \frac{t}{\lambda_i} \end{aligned}$$

For the exponential model, the hazard function $h(t)$ is constant over time. The Weibull model and lognormal models allow the hazard function to vary as a function of elapsed time (see and respectively).

- The *systematic component* λ_i is modeled as

$$\lambda_i = \exp(x_i \beta),$$

where x_i is the vector of explanatory variables, and β is the vector of coefficients.

Quantities of Interest

- The expected values (qi$ev) for the exponential model are simulations of the expected duration given x_i and draws of β from its posterior,

$$E(Y) = \lambda_i = \exp(x_i \beta).$$

- The predicted values (qi$pr) are draws from the exponential distribution with rate equal to the expected value.
- The first difference (or difference in expected values, qi$ev.diff), is

$$\text{FD} \;=\; E(Y \mid x_1) - E(Y \mid x),$$

where x and x_1 are different vectors of values for the explanatory variables.

- In conditional prediction models, the average expected treatment effect (att.ev) for the treatment group is

$$\frac{1}{\sum_{i=1}^n t_i} \sum_{i:t_i=1}^n \left\{ Y_i(t_i = 1) - E[Y_i(t_i = 0)] \right\},$$

where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. When $Y_i(t_i = 1)$ is censored rather than observed, we replace it with a simulation from the model given available knowledge of the censoring process. Variation in the simulations is due to two factors: uncertainty in the imputation process for censored y_i^* and uncertainty in simulating $E[Y_i(t_i = 0)]$, the counterfactual expected value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.

- In conditional prediction models, the average predicted treatment effect (att.pr) for the treatment group is

$$\frac{1}{\sum_{i=1}^{n} t_i} \sum_{i:t_i=1}^{n} \left\{ Y_i(t_i = 1) - Y_i(\widehat{t_i = 0}) \right\},$$

  where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. When $Y_i(t_i = 1)$ is censored rather than observed, we replace it with a simulation from the model given available knowledge of the censoring process. Variation in the simulations is due to two factors: uncertainty in the imputation process for censored y_i^* and uncertainty in simulating $Y_i(\widehat{t_i = 0})$, the counterfactual predicted value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.

Output Values

The output of each Zelig command contains useful information which you may view. For example, if you run `z.out <- zelig(Surv(Y, C) ~ X, model = exp, data)`, then you may examine the available information in `z.out` by using `names(z.out)`, see the coefficients by using z.out$coefficients, and a default summary of information through `summary(z.out)`.

See also

The exponential function is part of the survival library by Terry Therneau, ported to R by Thomas Lumley. Advanced users may wish to refer to `help(survfit)` in the survival library.

zelig-gamma

Gamma Regression for Continuous, Positive Dependent Variables

Use the gamma regression model if you have a positive-valued dependent variable such as the number of years a parliamentary cabinet endures, or the seconds you can stay airborne while jumping. The gamma distribution assumes that all waiting times are complete by the end of the study (censoring is not allowed).

Syntax

With reference classes:

```
z5 <- zgamma$new()
z5$zelig(Y ~ X1 + X ~ X, data = mydata)
z5$setx()
z5$sim()
```

With the Zelig 4 compatibility wrappers:

```
z.out <- zelig(Y ~ X1 + X2, model = "gamma", data = mydata)
x.out <- setx(z.out)
s.out <- sim(z.out, x = x.out, x1 = NULL)
```

Example

Attach the sample data:

```
data(coalition)
```

Estimate the model:

```
z.out <- zelig(duration ~ fract + numst2, model = "gamma", data = coalition)

## How to cite this model in Zelig:
##   Kosuke Imai, Gary King, Olivia Lau. 2007.
##   gamma: Gamma Regression for Continuous, Positive Dependent Variables
##   in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
##   http://datascience.iq.harvard.edu/zelig
```

View the regression output:

```
summary(z.out)

## Model: 1
## Call:  stats::glm(formula = duration ~ fract + numst2, family = Gamma("inverse"),
##     data = .)
##
## Coefficients:
## (Intercept)        fract       numst2
##   -0.012960     0.000115    -0.017387
##
## Degrees of Freedom: 313 Total (i.e. Null);  311 Residual
## Null Deviance:       301
## Residual Deviance: 272   AIC: 2430
## Next step: Use 'setx' method
```

Set the baseline values (with the ruling coalition in the minority) and the alternative values (with the ruling coalition in the majority) for X:

```
x.low <- setx(z.out, numst2 = 0)
x.high <- setx(z.out, numst2 = 1)
```

Simulate expected values (qi$ev) and first differences (qi$fd):

```
s.out <- sim(z.out, x = x.low, x1 = x.high)

summary(s.out)

##
##  sim x :
##  -----
## ev
##       mean    sd   50%  2.5% 97.5%
## [1,] 14.43 1.122 14.33 12.51 16.97
## pv
##       mean    sd   50%  2.5% 97.5%
## [1,] 14.01 12.42 10.22 0.613  46.1
##
##  sim x1 :
##  -----
## ev
##       mean    sd   50%  2.5% 97.5%
## [1,] 19.22 1.094 19.18 17.22 21.58
## pv
##       mean    sd   50%   2.5% 97.5%
## [1,] 19.21 17.47 14.08 0.7479 64.43
```

```
## fd
##       mean   sd   50%  2.5% 97.5%
## [1,] 4.789 1.55 4.821 1.772 7.908
```

```
plot(s.out)
```

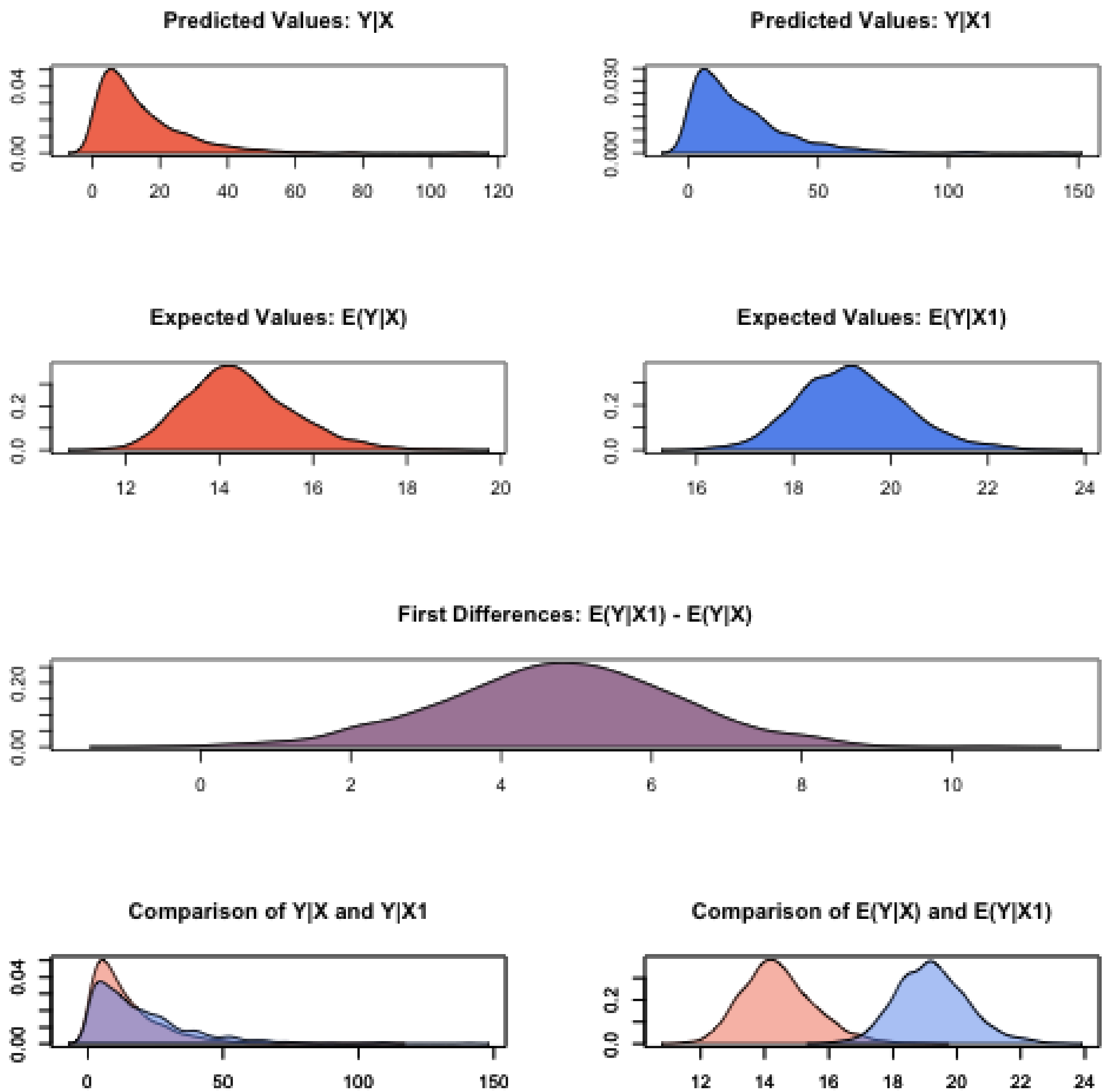


Figure 2.13: Zelig-gamma

Model

- The Gamma distribution with scale parameter α has a *stochastic component*:

$$\begin{aligned} Y \sim & \quad \text{Gamma}(y_i \mid \lambda_i, \alpha) \\ f(y) = & \quad \frac{1}{\alpha^{\lambda_i} \, \Gamma \lambda_i} \, y_i^{\lambda_i - 1} \exp - \left\{ \frac{y_i}{\alpha} \right\} \end{aligned}$$

  for $\alpha, \lambda_i, y_i > 0$.

- The *systematic component* is given by

$$\lambda_i = \frac{1}{x_i \beta}$$

Quantities of Interest

- The expected values (qi$ev) are simulations of the mean of the stochastic component given draws of α and β from their posteriors:

$$E(Y) = \alpha \lambda_i.$$

- The predicted values (qi$pr) are draws from the gamma distribution for each given set of parameters (α, λ_i).
- If x1 is specified, sim() also returns the differences in the expected values (qi$fd),

$$E(Y \mid x_1) - E(Y \mid x)$$

  .

- In conditional prediction models, the average expected treatment effect (att.ev) for the treatment group is

$$\frac{1}{\sum_{i=1}^n t_i} \sum_{i:t_i=1}^n \left\{ Y_i(t_i = 1) - E[Y_i(t_i = 0)] \right\},$$

  where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. Variation in the simulations are due to uncertainty in simulating $E[Y_i(t_i = 0)]$, the counterfactual expected value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.

- In conditional prediction models, the average predicted treatment effect (att.pr) for the treatment group is

$$\frac{1}{\sum_{i=1}^n t_i} \sum_{i:t_i=1}^n \left\{ Y_i(t_i = 1) - Y_i(\widehat{t_i = 0}) \right\},$$

  where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. Variation in the simulations are due to uncertainty in simulating $Y_i(\widehat{t_i = 0})$, the counterfactual predicted value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.

Output Values

The output of each Zelig command contains useful information which you may view. For example, if you run `z.out <- zelig(y ~ x, model = gamma, data)`, then you may examine the available information in `z.out` by using `names(z.out)`, see the coefficients by using z.out$coefficients, and a default summary of information through `summary(z.out)`.

See also

The gamma model is part of the stats package. Advanced users may wish to refer to `help(glm)` and `help(family)`.

zelig-logit

Logistic Regression for Dichotomous Dependent Variables

Logistic regression specifies a dichotomous dependent variable as a function of a set of explanatory variables.

Syntax

With reference classes:

```
z5 <- zlogit$new()
z5$zelig(Y ~ X1 + X ~ X, data = mydata)
z5$setx()
z5$sim()
```

With the Zelig 4 compatibility wrappers:

```
z.out <- zelig(Y ~ X1 + X2, model = "logit", data = mydata)
x.out <- setx(z.out)
s.out <- sim(z.out, x = x.out, x1 = NULL)
```

Examples

Basic Example Attaching the sample turnout dataset:

```
data(turnout)
```

Estimating parameter values for the logistic regression:

```
z.out1 <- zelig(vote ~ age + race, model = "logit", data = turnout)

## How to cite this model in Zelig:
##   Kosuke Imai, Gary King, Olivia Lau. 2007.
##   logit: Logistic Regression for Dichotomous Dependent Variables
##   in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
##   http://datascience.iq.harvard.edu/zelig
```

Setting values for the explanatory variables:

```
x.out1 <- setx(z.out1, age = 36, race = "white")
```

Simulating quantities of interest from the posterior distribution.

```
s.out1 <- sim(z.out1, x = x.out1)

summary(s.out1)

##
##  sim x :
##  -----
## ev
```

```
##        mean      sd    50%   2.5% 97.5%
## [1,] 0.7481 0.01149 0.7482 0.7264  0.77
## pv
##          0     1
## [1,] 0.265 0.735
```

```
plot(s.out1)
```

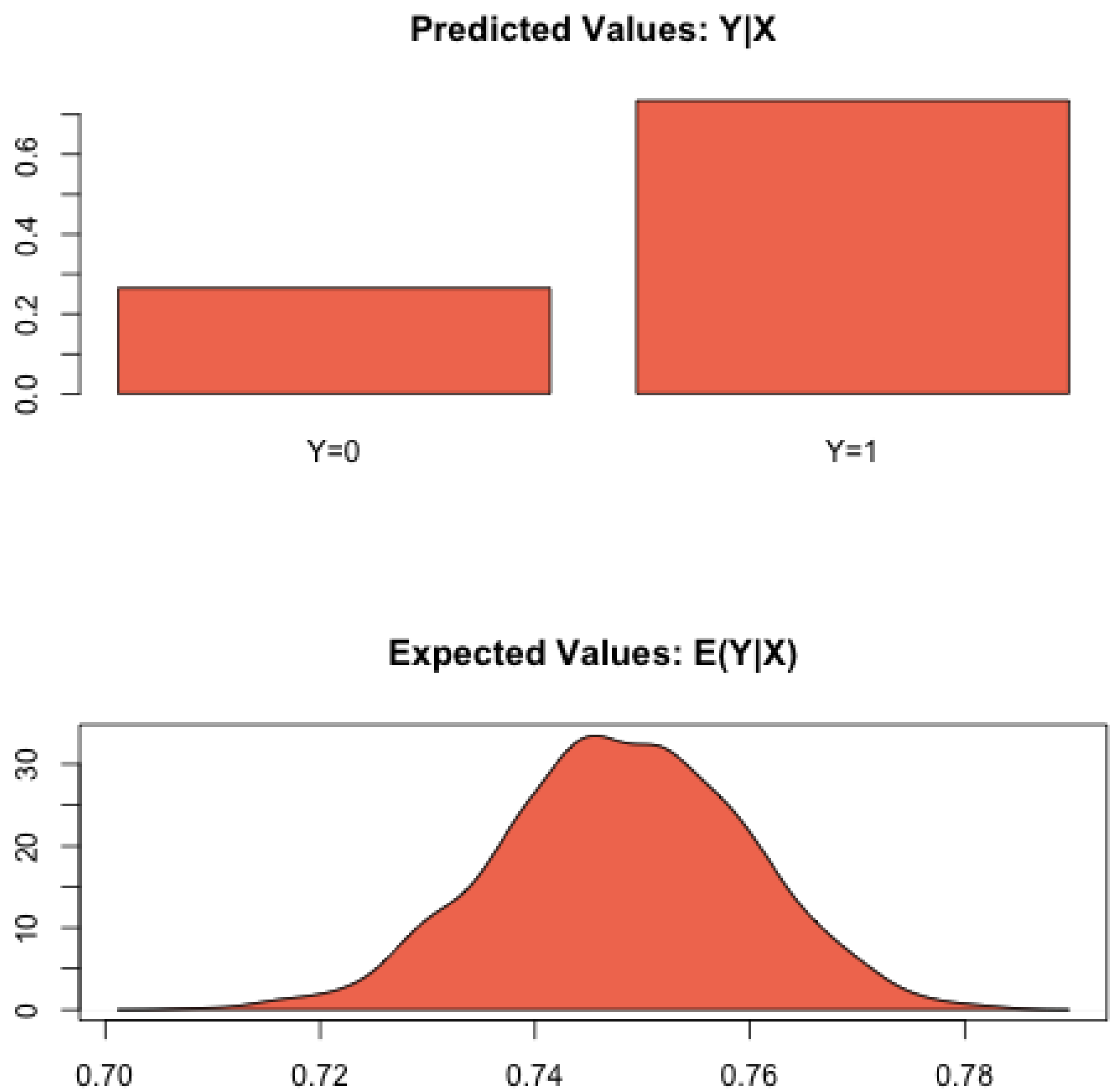


Figure 2.14: Zelig-logit-1

Simulating First Differences Estimating the risk difference (and risk ratio) between low education (25th percentile) and high education (75th percentile) while all the other variables held at their default values.

```
z.out2 <- zelig(vote ~ race + educate, model = "logit", data = turnout)

## How to cite this model in Zelig:
##   Kosuke Imai, Gary King, Olivia Lau. 2007.
##   logit: Logistic Regression for Dichotomous Dependent Variables
##   in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
##   http://datascience.iq.harvard.edu/zelig

x.high <- setx(z.out2, educate = quantile(turnout$educate, prob = 0.75))
x.low <- setx(z.out2, educate = quantile(turnout$educate, prob = 0.25))
s.out2 <- sim(z.out2, x = x.high, x1 = x.low)
summary(s.out2)

##
##  sim x :
##  -----
## ev
##        mean      sd   50%   2.5%  97.5%
## [1,] 0.8227 0.01035 0.823 0.8022 0.8413
## pv
##          0     1
## [1,] 0.166 0.834
##
##  sim x1 :
##  -----
## ev
##        mean      sd    50%   2.5%  97.5%
## [1,] 0.7087 0.01261 0.7086 0.6844 0.7338
## pv
##          0     1
## [1,] 0.282 0.718
## fd
##         mean      sd     50%    2.5%    97.5%
## [1,] -0.1141 0.01133 -0.1139 -0.1362 -0.09217

plot(s.out2)
```

Model

Let Y_i be the binary dependent variable for observation i which takes the value of either 0 or 1.

- The *stochastic component* is given by

$$\begin{aligned} Y_i \sim & \quad \text{Bernoulli}(y_i \mid \pi_i) \\ = & \quad \pi_i^{y_i}(1-\pi_i)^{1-y_i} \end{aligned}$$

  where $\pi_i = \Pr(Y_i = 1)$.

- The *systematic component* is given by:

$$\pi_i \;=\; \frac{1}{1+\exp(-x_i\beta)}.$$

  where x_i is the vector of k explanatory variables for observation i and β is the vector of coefficients.

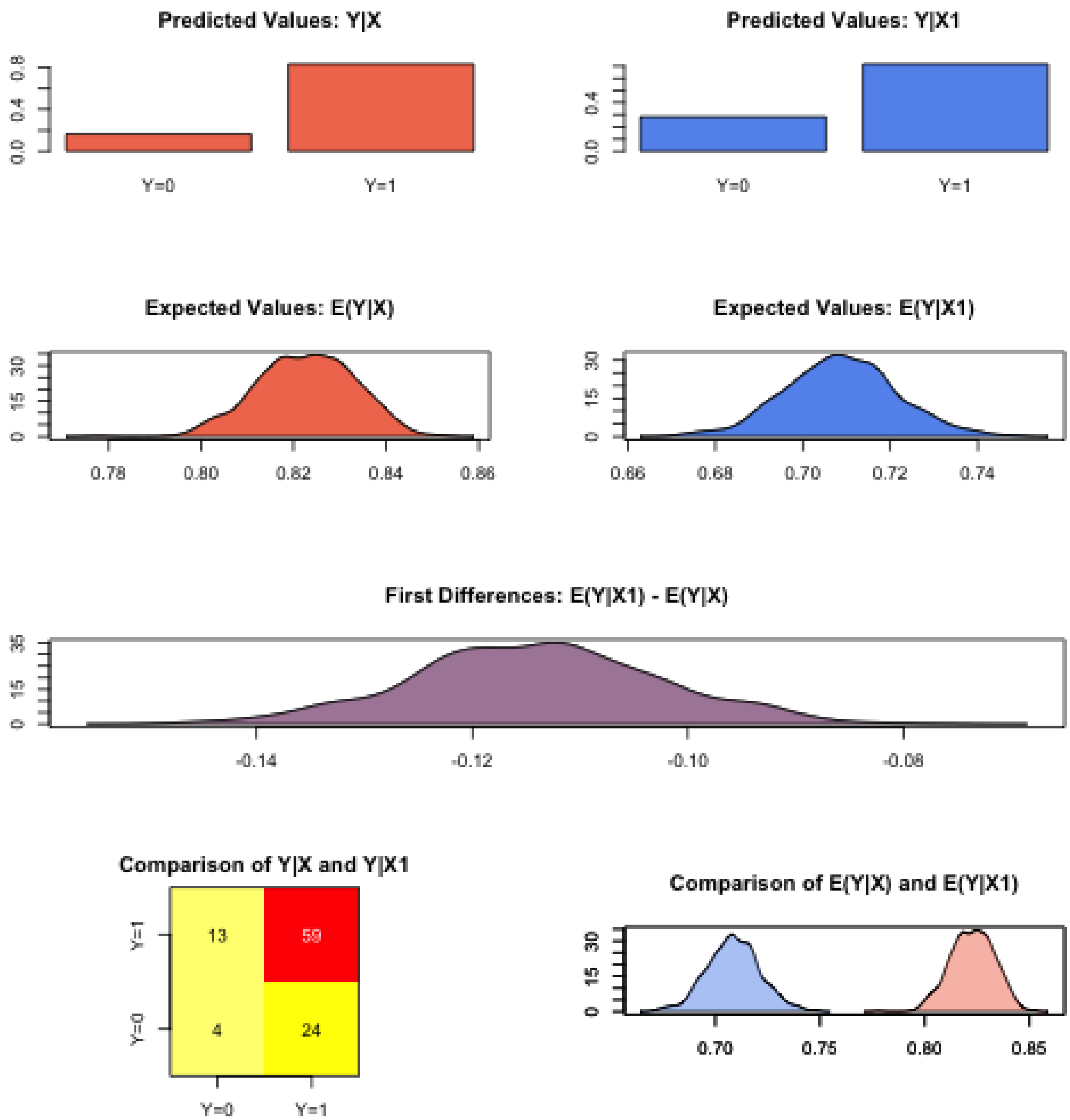


Figure 2.15: Zelig-logit-2

Quantities of Interest

- The expected values (qi$ev) for the logit model are simulations of the predicted probability of a success:

  $$E(Y) = \pi_i = \frac{1}{1 + \exp(-x_i \beta)},$$

  given draws of β from its sampling distribution.

- The predicted values (qi$pr) are draws from the Binomial distribution with mean equal to the simulated expected value π_i.

- The first difference (qi$fd) for the logit model is defined as

  $$\text{FD} = \Pr(Y = 1 \mid x_1) - \Pr(Y = 1 \mid x).$$

- The risk ratio (qi$rr) is defined as

  $$\text{RR} = \Pr(Y = 1 \mid x_1) \ / \ \Pr(Y = 1 \mid x).$$

- In conditional prediction models, the average expected treatment effect (att.ev) for the treatment group is

  $$\frac{1}{\sum_{i=1}^n t_i} \sum_{i:t_i=1}^n \left\{ Y_i(t_i = 1) - E[Y_i(t_i = 0)] \right\},$$

  where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. Variation in the simulations are due to uncertainty in simulating $E[Y_i(t_i = 0)]$, the counterfactual expected value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.

- In conditional prediction models, the average predicted treatment effect (att.pr) for the treatment group is

  $$\frac{1}{\sum_{i=1}^n t_i} \sum_{i:t_i=1}^n \left\{ Y_i(t_i = 1) - Y_i(\widehat{t_i = 0}) \right\},$$

  where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. Variation in the simulations are due to uncertainty in simulating $Y_i(\widehat{t_i = 0})$, the counterfactual predicted value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.

Output Values

The output of each Zelig command contains useful information which you may view. For example, if you run `z.out <- zelig(y ~ x, model = logit, data)`, then you may examine the available information in `z.out` by using `names(z.out)`, see the coefficients by using z.out$coefficients, and a default summary of information through `summary(z.out)`.

See also

The logit model is part of the stats package. Advanced users may wish to refer to `help(glm)` and `help(family)`.

zelig-lognorm

Log-Normal Regression for Duration Dependent Variables

The log-normal model describes an event's duration, the dependent variable, as a function of a set of explanatory variables. The log-normal model may take time censored dependent variables, and allows the hazard rate to increase and decrease.

Syntax

With reference classes:

```
z5 <- zlognorm$new()
z5$zelig(Surv(Y, C) ~ X, data = mydata)
z5$setx()
z5$sim()
```

With reference classes:

```
z5 <- zlognorm$new()
z5$zelig(Surv(Y, C) ~ X, data = mydata)
z5$setx()
z5$sim()
```

With the Zelig 4 compatibility wrappers:

```
z.out <- zelig(Surv(Y, C) ~ X, model = "lognorm", data = mydata)
x.out <- setx(z.out)
s.out <- sim(z.out, x = x.out)
```

Log-normal models require that the dependent variable be in the form Surv(Y, C), where Y and C are vectors of length n. For each observation i in 1, . . . , n, the value y_i is the duration (lifetime, for example) of each subject, and the associated c_i is a binary variable such that $c_i = 1$ if the duration is not censored (*e.g.*, the subject dies during the study) or $c_i = 0$ if the duration is censored (*e.g.*, the subject is still alive at the end of the study). If c_i is omitted, all Y are assumed to be completed; that is, time defaults to 1 for all observations.

Input Values

In addition to the standard inputs, zelig() takes the following additional options for lognormal regression:

- robust: defaults to FALSE. If TRUE, zelig() computes robust standard errors based on sandwich estimators (see and) based on the options in cluster.
- cluster: if robust = TRUE, you may select a variable to define groups of correlated observations. Let x3 be a variable that consists of either discrete numeric values, character strings, or factors that define strata. Then

```
z.out <- zelig(y ~ x1 + x2, robust = TRUE, cluster = "x3", model = "exp", data = mydata)
```

means that the observations can be correlated within the strata defined by the variable x3, and that robust standard errors should be calculated according to those clusters. If robust = TRUE but cluster is not specified, zelig() assumes that each observation falls into its own cluster.

Example

Attach the sample data:

```
data(coalition)
```

Estimate the model:

```
z.out <- zelig(Surv(duration, ciep12) ~ fract + numst2, model ="lognorm",  data = coalition)
```

```
## How to cite this model in Zelig:
##   Matthew Owen, Olivia Lau, Kosuke Imai, Gary King. 2007.
##   lognorm: Log-Normal Regression for Duration Dependent Variables
##   in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
##   http://datascience.iq.harvard.edu/zelig
```

View the regression output:

```
summary(z.out)
```

```
## Model: 1Call:
## survival::survreg(formula = Surv(duration, ciep12) ~ fract +
##     numst2, data = ., dist = "lognormal", model = FALSE)
##
## Coefficients:
## (Intercept)       fract      numst2
##    5.366670   -0.004438    0.559833
##
## Scale= 1.2
##
## Loglik(model)= -1078   Loglik(intercept only)= -1101
##  Chisq= 46.58 on 2 degrees of freedom, p= 7.7e-11
## n= 314
## Next step: Use 'setx' method
```

Set the baseline values (with the ruling coalition in the minority) and the alternative values (with the ruling coalition in the majority) for X:

```
x.low <- setx(z.out, numst2 = 0)
x.high <- setx(z.out, numst2= 1)
```

Simulate expected values (qi$ev) and first differences (qi$fd):

```
s.out <- sim(z.out, x = x.low, x1 = x.high)
```

```
summary(s.out)
```

```
##
##  sim x :
##  -----
## ev
##   mean   sd   50%  2.5% 97.5%
## 1 18.4 2.41 18.37 14.05 23.34
## pv
##   mean   sd   50%  2.5% 97.5%
## 1 18.4 2.41 18.37 14.05 23.34
##
##  sim x1 :
##  -----
## ev
##    mean    sd   50% 2.5% 97.5%
## 1 32.26 3.699 31.99 25.9 40.29
## pv
```

```
##    mean    sd   50% 2.5% 97.5%
## 1 32.26 3.699 31.99 25.9 40.29
## fd
##    mean    sd   50%  2.5% 97.5%
## 1 13.86 3.612 13.76 6.925 21.12
```

```
plot(s.out)
```

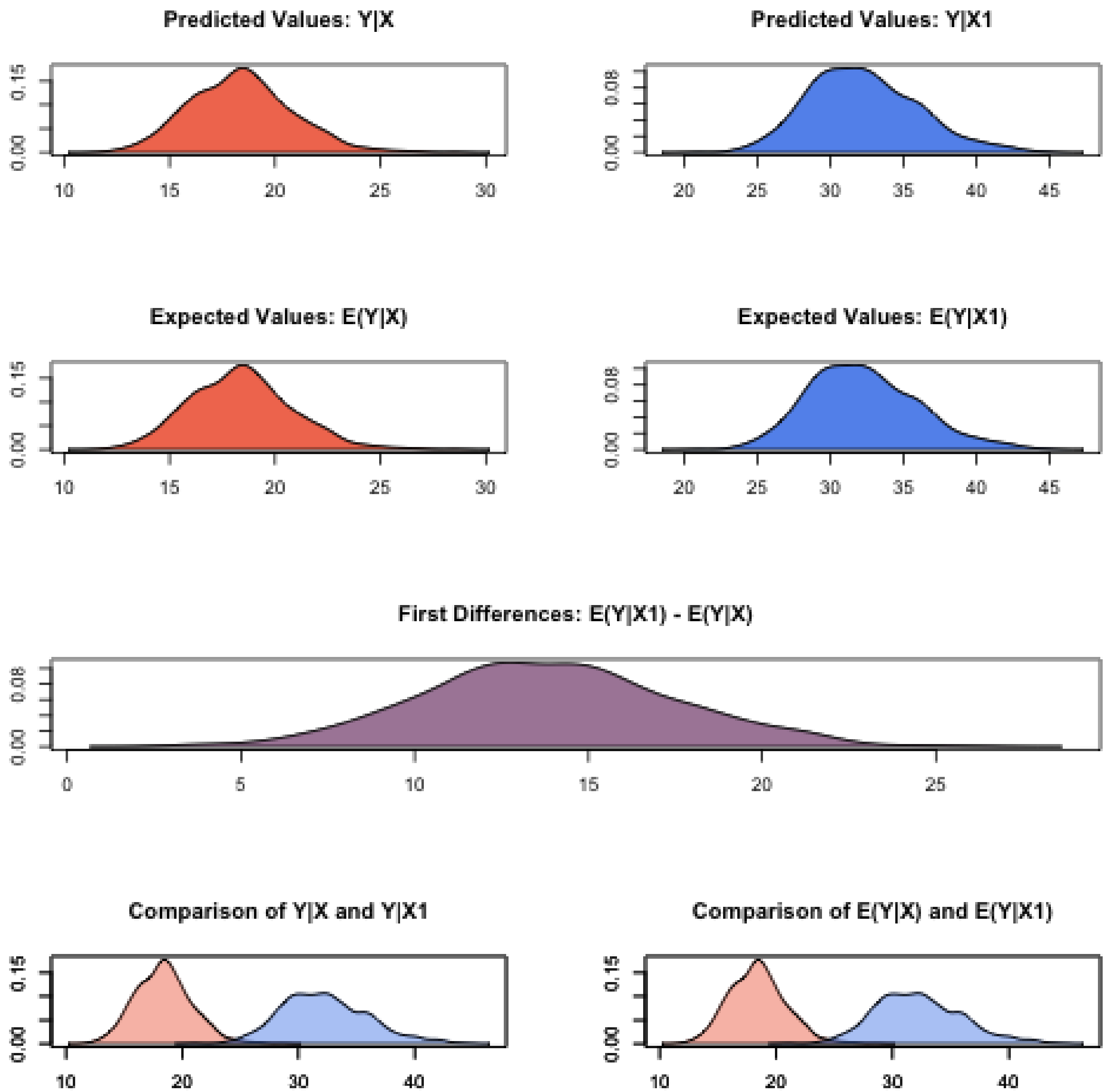


Figure 2.16: Zelig-lognorm

Model

Let Y_i^* be the survival time for observation i with the density function $f(y)$ and the corresponding distribution function $F(t) = \int_0^t f(y)dy$. This variable might be censored for some observations at a fixed time y_c such that the fully observed dependent variable, Y_i, is defined as

$$Y_i = \begin{cases} Y_i^* & \text{if } Y_i^* \leq y_c \\ y_c & \text{if } Y_i^* > y_c \end{cases}$$

- The *stochastic component* is described by the distribution of the partially observed variable, Y^*. For the lognormal model, there are two equivalent representations:

  $$Y_i^* \sim \text{LogNormal}(\mu_i, \sigma^2) \text{ or } \quad \log(Y_i^*) \sim \text{Normal}(\mu_i, \sigma^2)$$

  where the parameters μ_i and σ^2 are the mean and variance of the Normal distribution. (Note that the output from zelig() parameterizes scale:math:' = sigma'.)

  In addition, survival models like the lognormal have three additional properties. The hazard function $h(t)$ measures the probability of not surviving past time t given survival up to t. In general, the hazard function is equal to $f(t)/S(t)$ where the survival function $S(t) = 1 - \int_0^t f(s)ds$ represents the fraction still surviving at time t. The cumulative hazard function $H(t)$ describes the probability of dying before time t. In general, $H(t) = \int_0^t h(s)ds = -\log S(t)$. In the case of the lognormal model,

  $$\begin{aligned} h(t) = & \quad \frac{1}{\sqrt{2\pi}\,\sigma t\, S(t)} \exp\left\{-\frac{1}{2\sigma^2}(\log \lambda t)^2\right\} \\ S(t) = & \quad 1 - \Phi\left(\frac{1}{\sigma}\log \lambda t\right) \\ H(t) = & \quad -\log\left\{1 - \Phi\left(\frac{1}{\sigma}\log \lambda t\right)\right\} \end{aligned}$$

  where $\Phi(\cdot)$ is the cumulative density function for the Normal distribution.

- The *systematic component* is described as:

  $$\mu_i = x_i \beta.$$

Quantities of Interest

- The expected values (qi$ev) for the lognormal model are simulations of the expected duration:

  $$E(Y) = \exp\left(\mu_i + \frac{1}{2}\sigma^2\right),$$

  given draws of β and σ from their sampling distributions.

- The predicted value is a draw from the log-normal distribution given simulations of the parameters (λ_i, σ).
- The first difference (qi$fd) is

  $$\text{FD} = E(Y \mid x_1) - E(Y \mid x).$$

- In conditional prediction models, the average expected treatment effect (att.ev) for the treatment group is

  $$\frac{1}{\sum_{i=1}^n t_i} \sum_{i:t_i=1}^n \{Y_i(t_i = 1) - E[Y_i(t_i = 0)]\},$$

where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. When $Y_i(t_i = 1)$ is censored rather than observed, we replace it with a simulation from the model given available knowledge of the censoring process. Variation in the simulations is due to two factors: uncertainty in the imputation process for censored y_i^* and uncertainty in simulating $E[Y_i(t_i = 0)]$, the counterfactual expected value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.

- In conditional prediction models, the average predicted treatment effect (att.pr) for the treatment group is

$$\frac{1}{\sum_{i=1}^{n} t_i} \sum_{i:t_i=1}^{n} \{Y_i(t_i = 1) - Y_i(\widehat{t_i = 0})\},$$

where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. When $Y_i(t_i = 1)$ is censored rather than observed, we replace it with a simulation from the model given available knowledge of the censoring process. Variation in the simulations are due to two factors: uncertainty in the imputation process for censored y_i^* and uncertainty in simulating $Y_i(\widehat{t_i = 0})$, the counterfactual predicted value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.

Output Values

The output of each Zelig command contains useful information which you may view. For example, if you run `z.out <- zelig(Surv(Y, C) ~ X, model = lognorm, data)`, then you may examine the available information in `z.out` by using `names(z.out)`, see the coefficients by using z.out$coefficients, and a default summary of information through `summary(z.out)`.

See also

The exponential function is part of the survival library by by Terry Therneau, ported to R by Thomas Lumley. Advanced users may wish to refer to `help(survfit)` in the survival library.

zelig-ls

Least Squares Regression for Continuous Dependent Variables

Use least squares regression analysis to estimate the best linear predictor for the specified dependent variables.

Syntax

With reference classes:

```
z5 <- zls$new()
z5$zelig(Y ~ X1 + X ~ X, data = mydata)
z5$setx()
z5$sim()
```

With the Zelig 4 compatibility wrappers:

```
z.out <- zelig(Y ~ X1 + X2, model = "ls", data = mydata)
x.out <- setx(z.out)
s.out <- sim(z.out, x = x.out)
```

Examples

Basic Example with First Differences Attach sample data:

```
data(macro)
```

Estimate model:

```
z.out1 <- zelig(unem ~ gdp + capmob + trade, model = "ls", data = macro)

## How to cite this model in Zelig:
##   Kosuke Imai, Gary King, and Olivia Lau. 2007.
##   ls: Least Squares Regression for Continuous Dependent Variables
##   in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
##   http://datascience.iq.harvard.edu/zelig
```

Summarize regression coefficients:

```
summary(z.out1)

## Model: 1
## Call:
## stats::lm(formula = unem ~ gdp + capmob + trade, data = .)
##
## Coefficients:
## (Intercept)          gdp       capmob        trade
##      6.1813      -0.3236       1.4219       0.0199
##
## Next step: Use 'setx' method
```

Set explanatory variables to their default (mean/mode) values, with high (80th percentile) and low (20th percentile) values for the trade variable:

```
x.high <- setx(z.out1, trade = quantile(macro$trade, 0.8))
x.low <- setx(z.out1, trade = quantile(macro$trade, 0.2))
```

Generate first differences for the effect of high versus low trade on GDP:

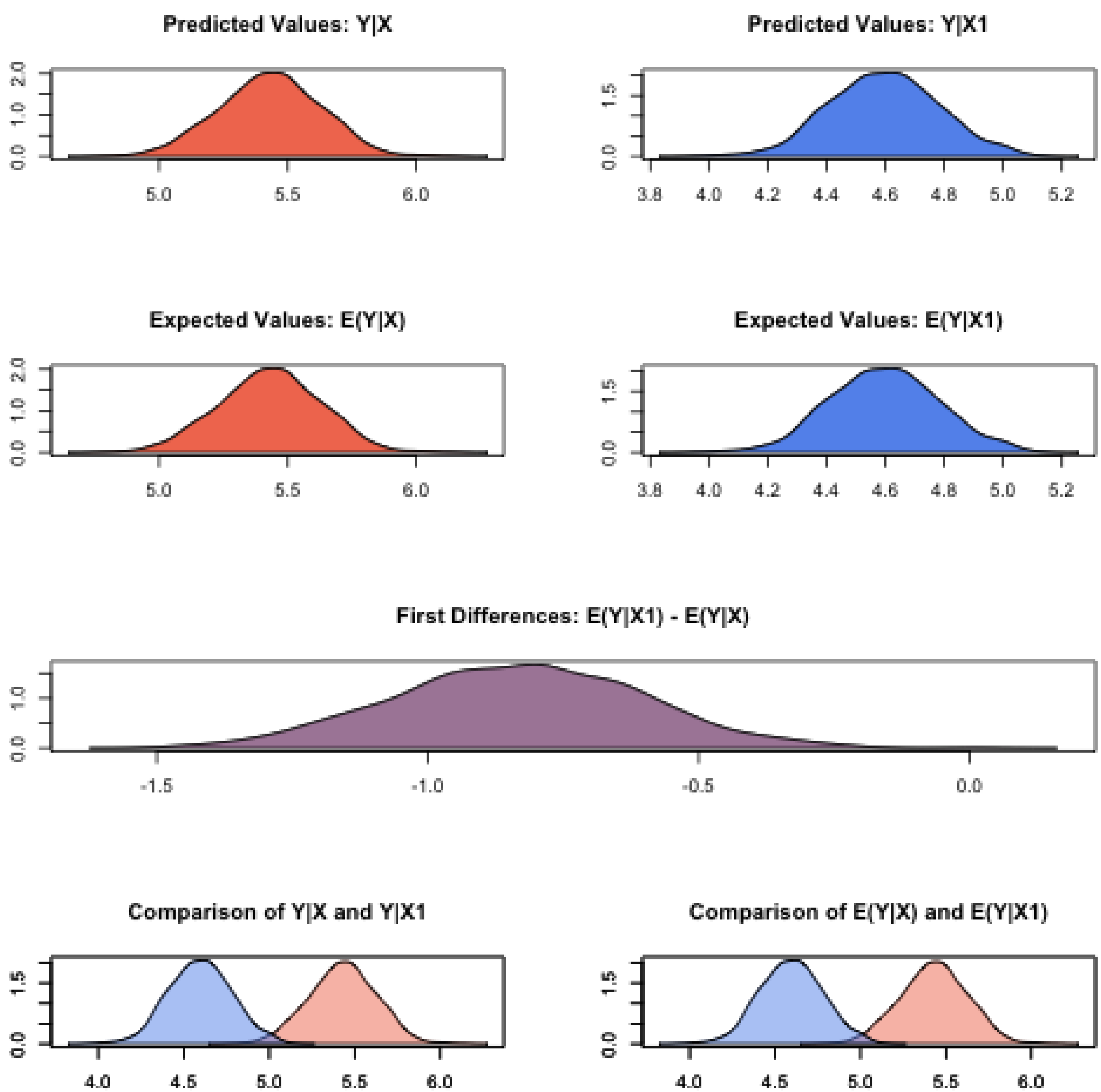
```
s.out1 <- sim(z.out1, x = x.high, x1 = x.low)

summary(s.out1)

##
##  sim x :
##  -----
## ev
##    mean     sd   50%  2.5% 97.5%
## 1 5.431 0.1964 5.432 5.056 5.796
## pv
##    mean     sd   50%  2.5% 97.5%
## 1 5.431 0.1964 5.432 5.056 5.796
##
##  sim x1 :
##  -----
## ev
##    mean     sd 50%  2.5% 97.5%
## 1 4.597 0.1838 4.6 4.245 4.969
## pv
##    mean     sd 50%  2.5% 97.5%
```

```
## 1 4.597 0.1838 4.6 4.245 4.969
## fd
##       mean     sd     50%   2.5%   97.5%
## 1 -0.8342 0.2281 -0.8266 -1.278 -0.3786
```

```
plot(s.out1)
```



Using Dummy Variables Estimate a model with fixed effects for each country (see for help with dummy variables). Note that you do not need to create dummy variables, as the program will automatically parse the unique values in the selected variable into discrete levels.

```
z.out2 <- zelig(unem ~ gdp + trade + capmob + as.factor(country), model = "ls", data = macro)
```

```
## How to cite this model in Zelig:
##   Kosuke Imai, Gary King, and Olivia Lau. 2007.
##   ls: Least Squares Regression for Continuous Dependent Variables
##   in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
##   http://datascience.iq.harvard.edu/zelig
```

Set values for the explanatory variables, using the default mean/mode values, with country set to the United States and Japan, respectively:

```
x.US <- setx(z.out2, country = "United States")
x.Japan <- setx(z.out2, country = "Japan")
```

Simulate quantities of interest:

```
s.out2 <- sim(z.out2, x = x.US, x1 = x.Japan)

plot(s.out2)
```

Model

- The *stochastic component* is described by a density with mean μ_i and the common variance σ^2

$$Y_i \sim f(y_i \mid \mu_i, \sigma^2).$$

- The *systematic component* models the conditional mean as

$$\mu_i = x_i \beta$$

  where x_i is the vector of covariates, and β is the vector of coefficients.

  The least squares estimator is the best linear predictor of a dependent variable given x_i, and minimizes the sum of squared residuals, $\sum_{i=1}^{n}(Y_i - x_i\beta)^2$.

Quantities of Interest

- The expected value (qi$ev) is the mean of simulations from the stochastic component,

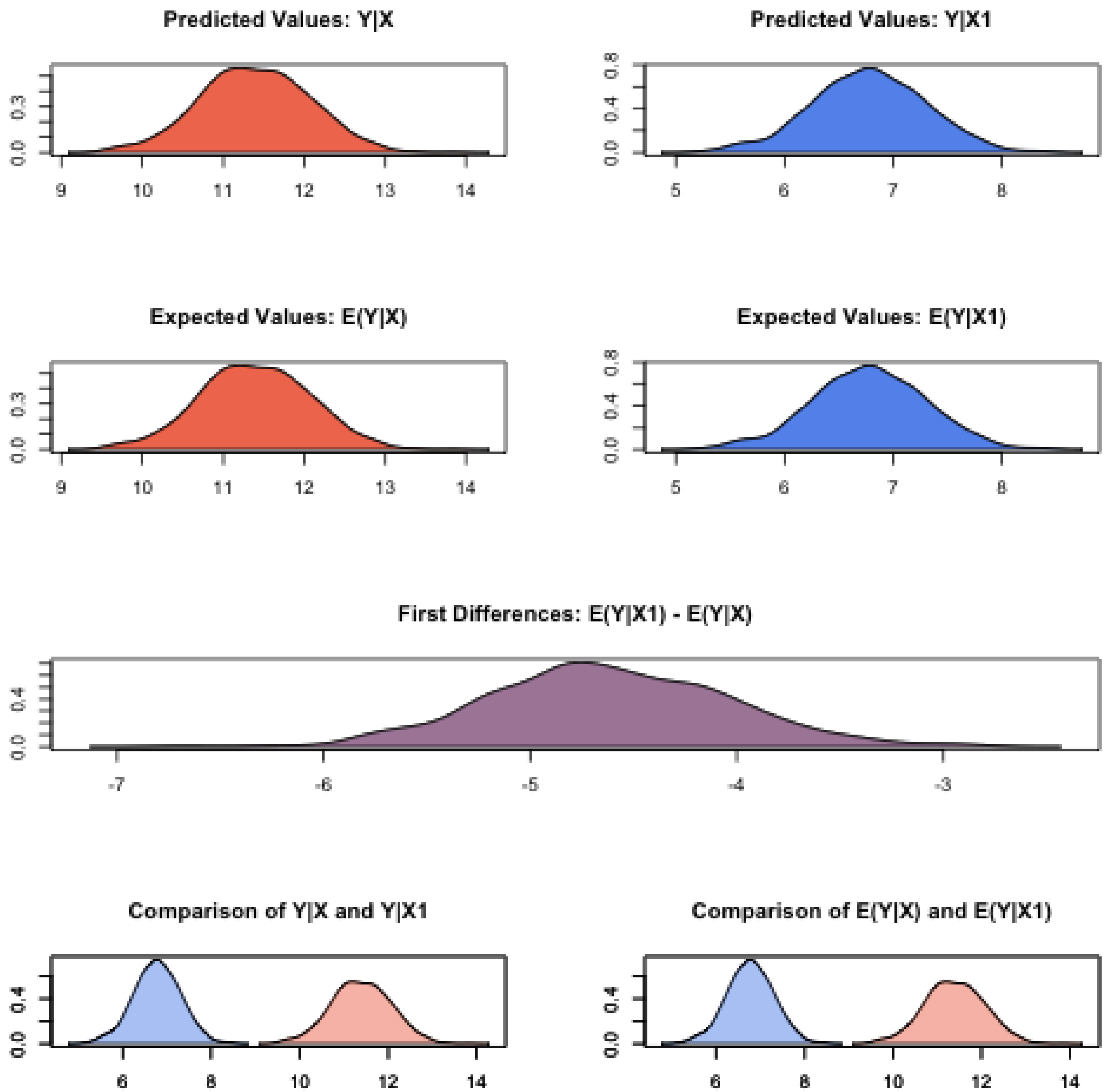
$$E(Y) = x_i\beta,$$

  given a draw of β from its sampling distribution.

- In conditional prediction models, the average expected treatment effect (att.ev) for the treatment group is

$$\frac{1}{\sum_{i=1}^{n} t_i} \sum_{i:t_i=1}^{n} \left\{ Y_i(t_i = 1) - E[Y_i(t_i = 0)] \right\},$$

  where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. Variation in the simulations are due to uncertainty in simulating $E[Y_i(t_i = 0)]$, the counterfactual expected value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.


Predicted Values: Y|X
Predicted Values: Y|X1
Expected Values: E(Y|X)
Expected Values: E(Y|X1)
First Differences: E(Y|X1) - E(Y|X)
Comparison of Y|X and Y|X1
Comparison of E(Y|X) and E(Y|X1)

Output Values

The output of each Zelig command contains useful information which you may view. For example, if you run `z.out <- zelig(y ~ x, model = ls, data)`, then you may examine the available information in `z.out` by using `names(z.out)`, see the coefficients by using z.out$coefficients, and a default summary of information through `summary(z.out)`. Other elements available through the $ operator are listed below.

- From the zelig() output object z.out, you may extract:
  - coefficients: parameter estimates for the explanatory variables.
  - residuals: the working residuals in the final iteration of the IWLS fit.
  - fitted.values: fitted values.
  - df.residual: the residual degrees of freedom.
  - zelig.data: the input data frame if save.data = TRUE.
- From summary(z.out), you may extract:
  - coefficients: the parameter estimates with their associated standard errors, p-values, and t-statistics.

$$\hat{\beta} = \left(\sum_{i=1}^{n} x_i' x_i \right)^{-1} \sum x_i y_i$$

  - sigma: the square root of the estimate variance of the random error e:

$$\hat{\sigma} = \frac{\sum (Y_i - x_i \hat{\beta})^2}{n - k}$$

  - r.squared: the fraction of the variance explained by the model.

$$R^2 = 1 - \frac{\sum (Y_i - x_i \hat{\beta})^2}{\sum (y_i - \bar{y})^2}$$

  - adj.r.squared: the above R^2 statistic, penalizing for an increased number of explanatory variables.
  - cov.unscaled: a $k \times k$ matrix of unscaled covariances.

See also

The least squares regression is part of the stats package by William N. Venables and Brian D. Ripley .In addition, advanced users may wish to refer to `help(lm)` and `help(lm.fit)`.

zelig-negbin

Negative Binomial Regression for Event Count Dependent Variables

Use the negative binomial regression if you have a count of events for each observation of your dependent variable. The negative binomial model is frequently used to estimate over-dispersed event count models.

Syntax

With reference classes:

```
z5 <- znegbin$new()
z5$zelig(Y ~ X1 + X ~ X, data = mydata)
z5$setx()
z5$sim()
```

With the Zelig 4 compatibility wrappers:

```
z.out <- zelig(Y ~ X1 + X2, model = "negbin", data = mydata)
x.out <- setx(z.out)
s.out <- sim(z.out, x = x.out)
```

Example

Load sample data:

```
data(sanction)
```

Estimate the model:

```
z.out <- zelig(num ~ target + coop, model = "negbinom", data = sanction)
```

```
## Error: Model 'negbinom' not found
```

```
summary(z.out)
```

```
## Model: 1Call:
## survival::survreg(formula = Surv(duration, ciep12) ~ fract +
##     numst2, data = ., dist = "lognormal", model = FALSE)
##
## Coefficients:
## (Intercept)       fract      numst2
##    5.366670   -0.004438    0.559833
##
## Scale= 1.2
##
## Loglik(model)= -1078   Loglik(intercept only)= -1101
##  Chisq= 46.58 on 2 degrees of freedom, p= 7.7e-11
## n= 314
## Next step: Use 'setx' method
```

Set values for the explanatory variables to their default mean values:

```
x.out <- setx(z.out)
```

Simulate fitted values:

```
s.out <- sim(z.out, x = x.out)
```

```
summary(s.out)
```

```
##
##  sim x :
##  -----
## ev
##    mean    sd   50%  2.5% 97.5%
## 1 26.05 2.643 25.86 21.33  31.8
## pv
```

```
##    mean    sd   50%  2.5% 97.5%
## 1 26.05 2.643 25.86 21.33  31.8
```

```
plot(s.out)
```

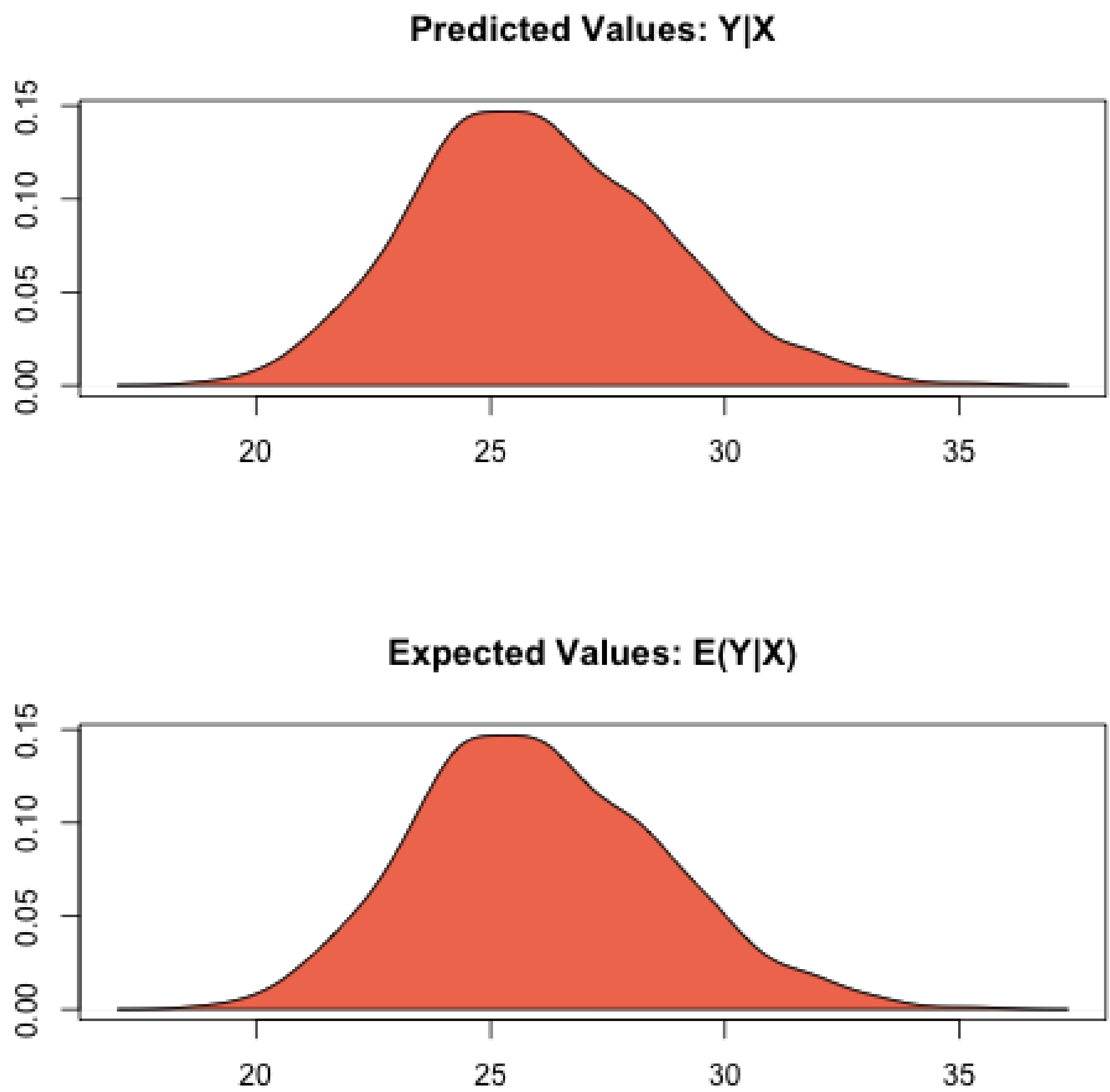


Figure 2.17: Zelig-negbin

Model

Let Y_i be the number of independent events that occur during a fixed time period. This variable can take any non-negative integer value.

- The negative binomial distribution is derived by letting the mean of the Poisson distribution vary according to a fixed parameter ζ given by the Gamma distribution. The *stochastic component* is given by

$$\begin{aligned} Y_i \mid \zeta_i &\sim \text{Poisson}(\zeta_i \mu_i), \\ \zeta_i &\sim \frac{1}{\theta}\text{Gamma}(\theta). \end{aligned}$$

The marginal distribution of Y_i is then the negative binomial with mean μ_i and variance $\mu_i + \mu_i^2/\theta$:

$$\begin{aligned} Y_i &\sim \text{NegBin}(\mu_i, \theta), \\ &= \frac{\Gamma(\theta + y_i)}{y!\,\Gamma(\theta)} \frac{\mu_i^{y_i}\,\theta^{\theta}}{(\mu_i + \theta)^{\theta + y_i}}, \end{aligned}$$

where θ is the systematic parameter of the Gamma distribution modeling ζ_i.

- The *systematic component* is given by

$$\mu_i = \exp(x_i \beta)$$

where x_i is the vector of k explanatory variables and β is the vector of coefficients.

Quantities of Interest

- The expected values (qi$ev) are simulations of the mean of the stochastic component. Thus,

$$E(Y) = \mu_i = \exp(x_i \beta),$$

given simulations of β.

- The predicted value (qi$pr) drawn from the distribution defined by the set of parameters (μ_i, θ).
- The first difference (qi$fd) is

$$\text{FD} \; = \; E(Y|x_1) - E(Y \mid x)$$

- In conditional prediction models, the average expected treatment effect (att.ev) for the treatment group is

$$\frac{1}{\sum_{i=1}^n t_i} \sum_{i:t_i=1}^{n} \left\{ Y_i(t_i = 1) - E[Y_i(t_i = 0)] \right\},$$

where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. Variation in the simulations are due to uncertainty in simulating $E[Y_i(t_i = 0)]$, the counterfactual expected value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.

- In conditional prediction models, the average predicted treatment effect (att.pr) for the treatment group is

$$\frac{1}{\sum_{i=1}^n t_i} \sum_{i:t_i=1}^{n} \left\{ Y_i(t_i = 1) - Y_i(\widehat{t_i = 0}) \right\},$$

where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. Variation in the simulations are due to uncertainty in simulating $Y_i(\widehat{t_i = 0})$, the counterfactual predicted value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.

Output Values

The output of each Zelig command contains useful information which you may view. For example, if you run `z.out <- zelig(y ~ x, model = negbin, data)`, then you may examine the available information in `z.out` by using `names(z.out)`, see the coefficients by using z.out$coefficients, and a default summary of information through `summary(z.out)`.

See also

The negative binomial model is part of the MASS package by William N. Venable and Brian D. Ripley . Advanced users may wish to refer to ‘‘help(glm.nb)‘.

zelig-normal

Normal Regression for Continuous Dependent Variables

The Normal regression model is a close variant of the more standard least squares regression model (see). Both models specify a continuous dependent variable as a linear function of a set of explanatory variables. The Normal model reports maximum likelihood (rather than least squares) estimates. The two models differ only in their estimate for the stochastic parameter σ.

Syntax

With reference classes:

```
z5 <- znormal$new()
z5$zelig(Y ~ X1 + X ~ X, data = mydata)
z5$setx()
z5$sim()
```

With the Zelig 4 compatibility wrappers:

```
z.out <- zelig(Y ~ X1 + X2, model = "normal", data = mydata)
x.out <- setx(z.out)
s.out <- sim(z.out, x = x.out)
```

Examples

Basic Example with First Differences Attach sample data:

```
data(macro)
```

Estimate model:

```
z.out1 <- zelig(unem ~ gdp + capmob + trade, model = "normal", data = macro)
```

```
## How to cite this model in Zelig:
##   Kosuke Imai, Gary King, Olivia Lau. 2008.
##   normal: Normal Regression for Continuous Dependent Variables
##   in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
##   http://datascience.iq.harvard.edu/zelig
```

Summarize of regression coefficients:

```
summary(z.out1)

## Model: 1
## Call:  stats::glm(formula = unem ~ gdp + capmob + trade, family = gaussian("identity"),
##     data = .)
##
## Coefficients:
## (Intercept)          gdp       capmob        trade
##      6.1813      -0.3236       1.4219       0.0199
##
## Degrees of Freedom: 349 Total (i.e. Null);  346 Residual
## Null Deviance:       3660
## Residual Deviance: 2610  AIC: 1710
## Next step: Use 'setx' method
```

Set explanatory variables to their default (mean/mode) values, with high (80th percentile) and low (20th percentile) values for trade:

```
x.high <- setx(z.out1, trade = quantile(macro$trade, 0.8))
x.low <- setx(z.out1, trade = quantile(macro$trade, 0.2))
```

Generate first differences for the effect of high versus low trade on GDP:

```
s.out1 <- sim(z.out1, x = x.high, x1 = x.low)

summary(s.out1)

##
##  sim x :
##  -----
## ev
##       mean     sd   50%  2.5% 97.5%
## [1,] 5.427 0.1905 5.426 5.066  5.78
## pv
##       mean    sd 50%   2.5% 97.5%
## [1,] 5.491 2.833 5.4 0.3017 11.19
##
##  sim x1 :
##  -----
## ev
##       mean     sd   50%  2.5% 97.5%
## [1,] 4.597 0.1845 4.599 4.238 4.979
## pv
##       mean    sd   50%   2.5% 97.5%
## [1,] 4.449 2.687 4.432 -1.007 9.682
## fd
##       mean     sd     50%   2.5%   97.5%
## [1,] -0.83 0.2307 -0.8362 -1.238 -0.3597
```

A visual summary of quantities of interest:

```
plot(s.out1)
```

Model

Let Y_i be the continuous dependent variable for observation i.

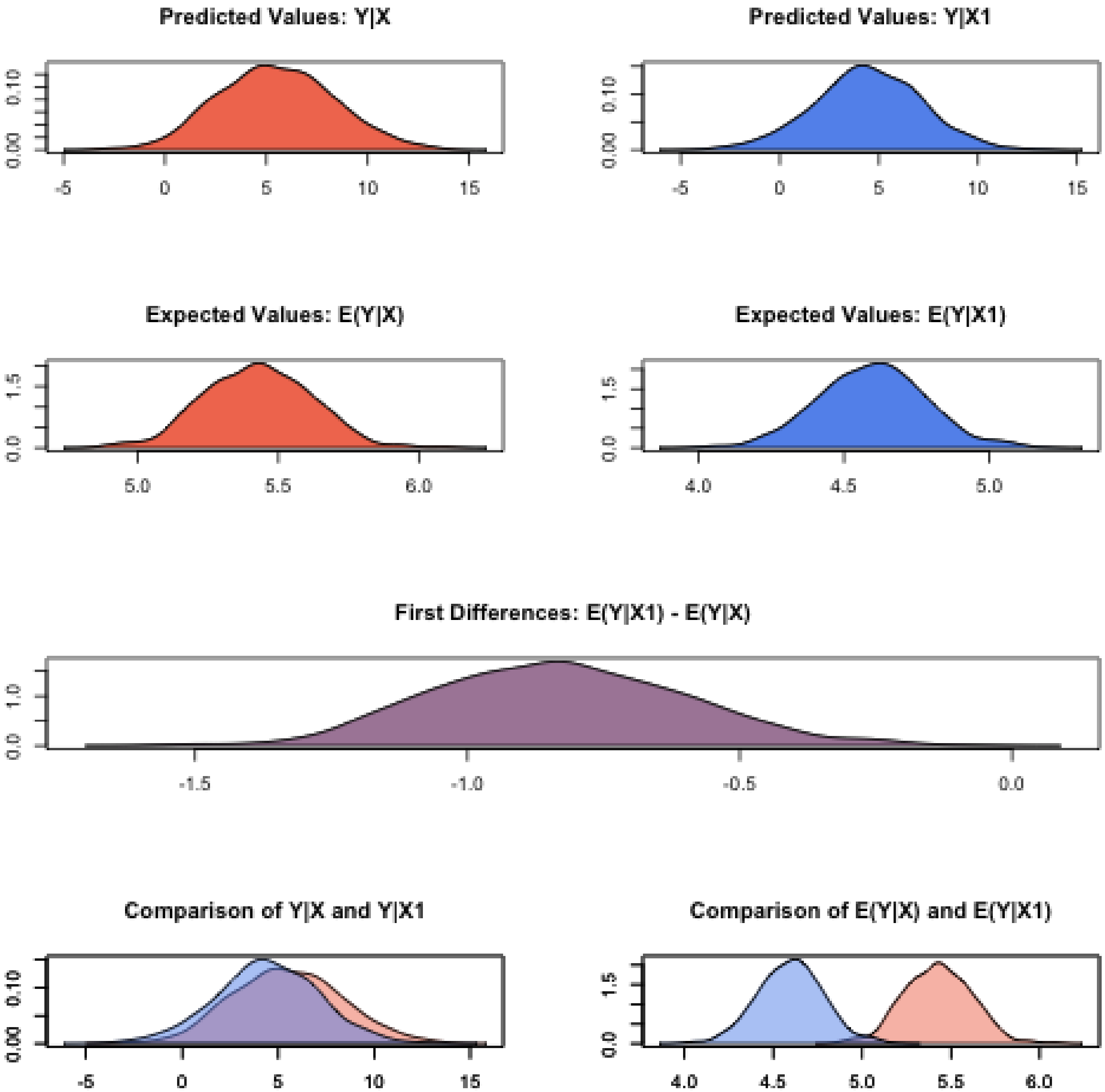


Figure 2.18: Zelig-normal

- The *stochastic component* is described by a univariate normal model with a vector of means μ_i and scalar variance σ^2:

$$Y_i \sim \text{Normal}(\mu_i, \sigma^2).$$

- The *systematic component* is

$$\mu_i = x_i \beta,$$

where x_i is the vector of k explanatory variables and β is the vector of coefficients.

Quantities of Interest

- The expected value (qi$ev) is the mean of simulations from the the stochastic component,

$$E(Y) = \mu_i = x_i \beta,$$

given a draw of β from its posterior.

- The predicted value (qi$pr) is drawn from the distribution defined by the set of parameters (μ_i, σ).
- The first difference (qi$fd) is:

$$\text{FD} = E(Y \mid x_1) - E(Y \mid x)$$

- In conditional prediction models, the average expected treatment effect (att.ev) for the treatment group is

$$\frac{1}{\sum_{i=1}^{n} t_i} \sum_{i:t_i=1}^{n} \{Y_i(t_i = 1) - E[Y_i(t_i = 0)]\},$$

where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. Variation in the simulations are due to uncertainty in simulating $E[Y_i(t_i = 0)]$, the counterfactual expected value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.

- In conditional prediction models, the average predicted treatment effect (att.pr) for the treatment group is

$$\frac{1}{\sum_{i=1}^{n} t_i} \sum_{i:t_i=1}^{n} \left\{ Y_i(t_i = 1) - Y_i(\widehat{t_i = 0}) \right\},$$

where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. Variation in the simulations are due to uncertainty in simulating $Y_i(\widehat{t_i = 0})$, the counterfactual predicted value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.

Output Values

The output of each Zelig command contains useful information which you may view. For example, if you run `z.out <- zelig(y ~ x, model = normal, data)`, then you may examine the available information in `z.out` by using `names(z.out)`, see the coefficients by using z.out$coefficients, and a default summary of information through `summary(z.out)`.

See also

The normal model is part of the stats package by . Advanced users may wish to refer to `help(glm)` and `help(family)`.

zelig-poisson

Poisson Regression for Event Count Dependent Variables

Use the Poisson regression model if the observations of your dependent variable represents the number of independent events that occur during a fixed period of time (see the negative binomial model, , for over-dispersed event counts.) For a Bayesian implementation of this model, see .

Syntax

With reference classes:

```
z5 <- zpoisson$new()
z5$zelig(Y ~ X1 + X ~ X, data = mydata)
z5$setx()
z5$sim()
```

With the Zelig 4 compatibility wrappers:

```
z.out <- zelig(Y ~ X1 + X2, model = "poisson", data = mydata)
x.out <- setx(z.out)
s.out <- sim(z.out, x = x.out)
```

Example

Load sample data:

```
data(sanction)
```

Estimate Poisson model:

```
z.out <- zelig(num ~ target + coop, model = "poisson", data = sanction)

## How to cite this model in Zelig:
##   Kosuke Imai, Gary King, Olivia Lau. 2007.
##   poisson: Poisson Regression for Event Count Dependent Variables
##   in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
##   http://datascience.iq.harvard.edu/zelig

summary(z.out)

## Model: 1
## Call:  stats::glm(formula = num ~ target + coop, family = poisson("log"),
##     data = .)
##
## Coefficients:
## (Intercept)       target         coop
##      -0.968       -0.021        1.211
##
## Degrees of Freedom: 77 Total (i.e. Null);  75 Residual
## Null Deviance:       1580
## Residual Deviance: 721   AIC: 944
## Next step: Use 'setx' method
```

Set values for the explanatory variables to their default mean values:

```
x.out <- setx(z.out)
```

Simulate fitted values:

```
s.out <- sim(z.out, x = x.out)
summary(s.out)
```

```
##
##  sim x :
##  -----
## ev
##       mean     sd  50%  2.5% 97.5%
## [1,] 3.256 0.2405 3.25 2.812 3.728
## pv
##       mean    sd 50% 2.5% 97.5%
## [1,] 3.206 1.878   3    0     7
```

```
plot(s.out)
```

Model

Let Y_i be the number of independent events that occur during a fixed time period. This variable can take any non-negative integer.

- The Poisson distribution has *stochastic component*

  $$Y_i \;\sim\; \textrm{Poisson}(\lambda_i),$$

  where λ_i is the mean and variance parameter.

- The *systematic component* is

  $$\lambda_i \;=\; \exp(x_i\beta),$$

  where x_i is the vector of explanatory variables, and β is the vector of coefficients.

Quantities of Interest

- The expected value (qi$ev) is the mean of simulations from the stochastic component,

  $$E(Y) = \lambda_i = \exp(x_i\beta),$$

  given draws of β from its sampling distribution.

- The predicted value (qi$pr) is a random draw from the poisson distribution defined by mean λ_i.

- The first difference in the expected values (qi$fd) is given by:

  $$\textrm{FD} \;=\; E(Y|x_1) - E(Y \mid x)$$

- In conditional prediction models, the average expected treatment effect (att.ev) for the treatment group is

  $$\frac{1}{\sum_{i=1}^n t_i}\sum_{i:t_i=1}^n \left\{ Y_i(t_i=1) - E[Y_i(t_i=0)] \right\},$$

  where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. Variation in the simulations are due to uncertainty in simulating $E[Y_i(t_i = 0)]$, the counterfactual expected value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.

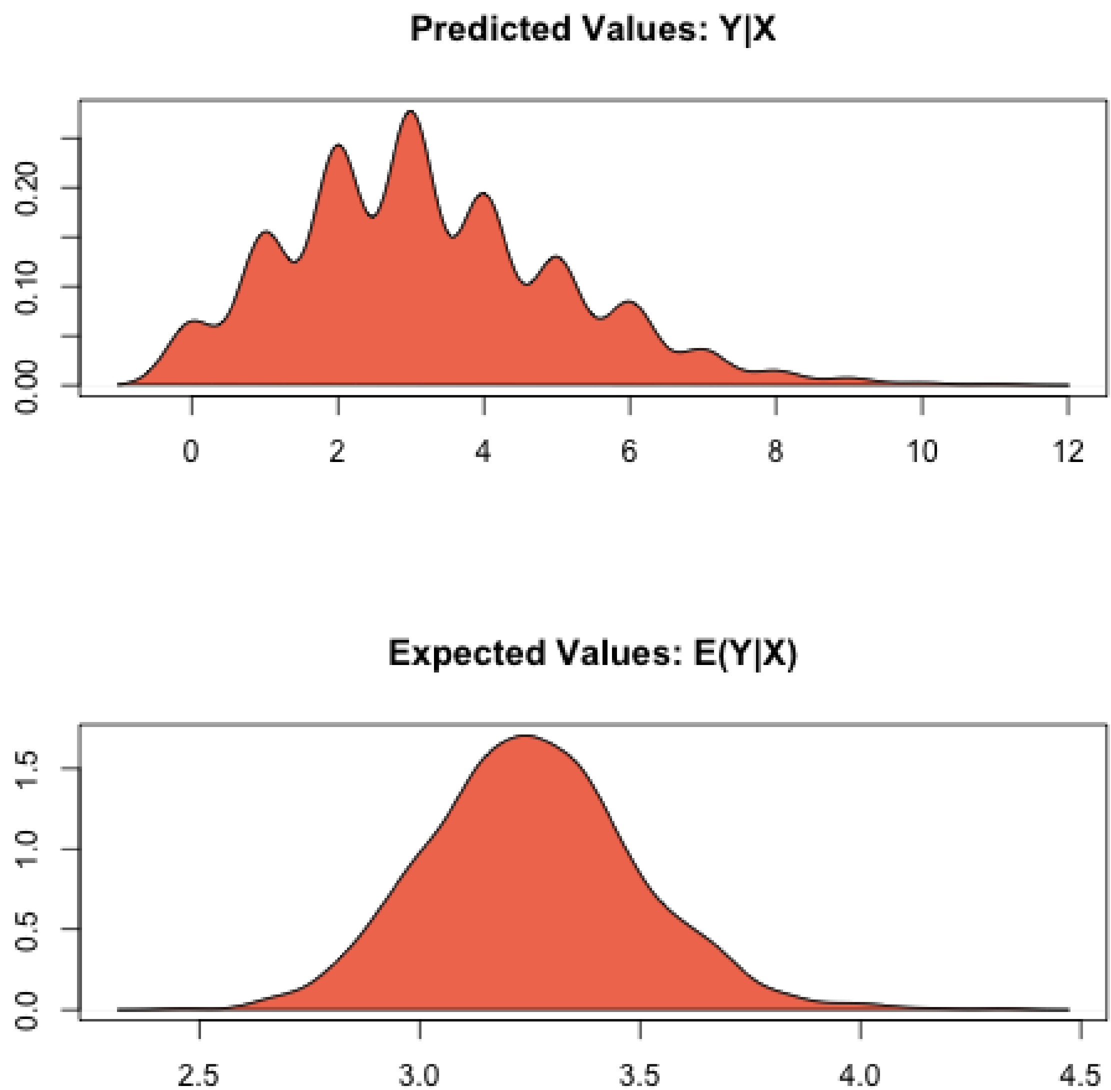


Figure 2.19: Zelig-poisson

- In conditional prediction models, the average predicted treatment effect (att.pr) for the treatment group is

$$\frac{1}{\sum_{i=1}^{n} t_i} \sum_{i:t_i=1}^{n} \left\{ Y_i(t_i = 1) - Y_i(\widehat{t_i = 0}) \right\},$$

  where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. Variation in the simulations are due to uncertainty in simulating $Y_i(\widehat{t_i = 0})$, the counterfactual predicted value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.

Output Values

The output of each Zelig command contains useful information which you may view. For example, if you run `z.out <- zelig(y ~ x, model = poisson, data)`, then you may examine the available information in `z.out` by using `names(z.out)`, see the coefficients by using z.out$coefficients, and a default summary of information through `summary(z.out)`.

See also

The poisson model is part of the stats package by . Advanced users may wish to refer to `help(glm)` and `help(family)`.

zelig-probit

Probit Regression for Dichotomous Dependent Variables

Use probit regression to model binary dependent variables specified as a function of a set of explanatory variables.

Syntax

With reference classes:

```
z5 <- zprobit$new()
z5$zelig(Y ~ X1 + X ~ X, data = mydata)
z5$setx()
z5$sim()
```

With the Zelig 4 compatibility wrappers:

```
z.out <- zelig(Y ~ X1 + X2, model = "probit", data = mydata)
x.out <- setx(z.out)
s.out <- sim(z.out, x = x.out, x1 = NULL)
```

Example

Attach the sample turnout dataset:

```
data(turnout)
```

Estimate parameter values for the probit regression:

```
z.out <- zelig(vote ~ race + educate, model = "probit", data = turnout)

## How to cite this model in Zelig:
##   Kosuke Imai, Gary King, Olivia Lau. 2007.
##   probit: Probit Regression for Dichotomous Dependent Variables
##   in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
##   http://datascience.iq.harvard.edu/zelig

summary(z.out)

## Model: 1
## Call:  stats::glm(formula = vote ~ race + educate, family = binomial("probit"),
##     data = .)
##
## Coefficients:
## (Intercept)    racewhite      educate
##     -0.7259       0.2991       0.0971
##
## Degrees of Freedom: 1999 Total (i.e. Null);  1997 Residual
## Null Deviance:       2270
## Residual Deviance: 2140  AIC: 2140
## Next step: Use 'setx' method
```

Set values for the explanatory variables to their default values.

```
x.out <- setx(z.out)
```

Simulate quantities of interest from the posterior distribution.

```
s.out <- sim(z.out, x = x.out)

summary(s.out)

plot(s.out1)
```

Model

Let Y_i be the observed binary dependent variable for observation i which takes the value of either 0 or 1.

- The *stochastic component* is given by

$$Y_i \sim \text{Bernoulli}(\pi_i),$$

where $\pi_i = \Pr(Y_i = 1)$.

- The *systematic component* is

$$\pi_i = \Phi(x_i \beta)$$

where $\Phi(\mu)$ is the cumulative distribution function of the Normal distribution with mean 0 and unit variance.

Quantities of Interest

- The expected value (qi$ev) is a simulation of predicted probability of success

$$E(Y) = \pi_i = \Phi(x_i \beta),$$

given a draw of β from its sampling distribution.

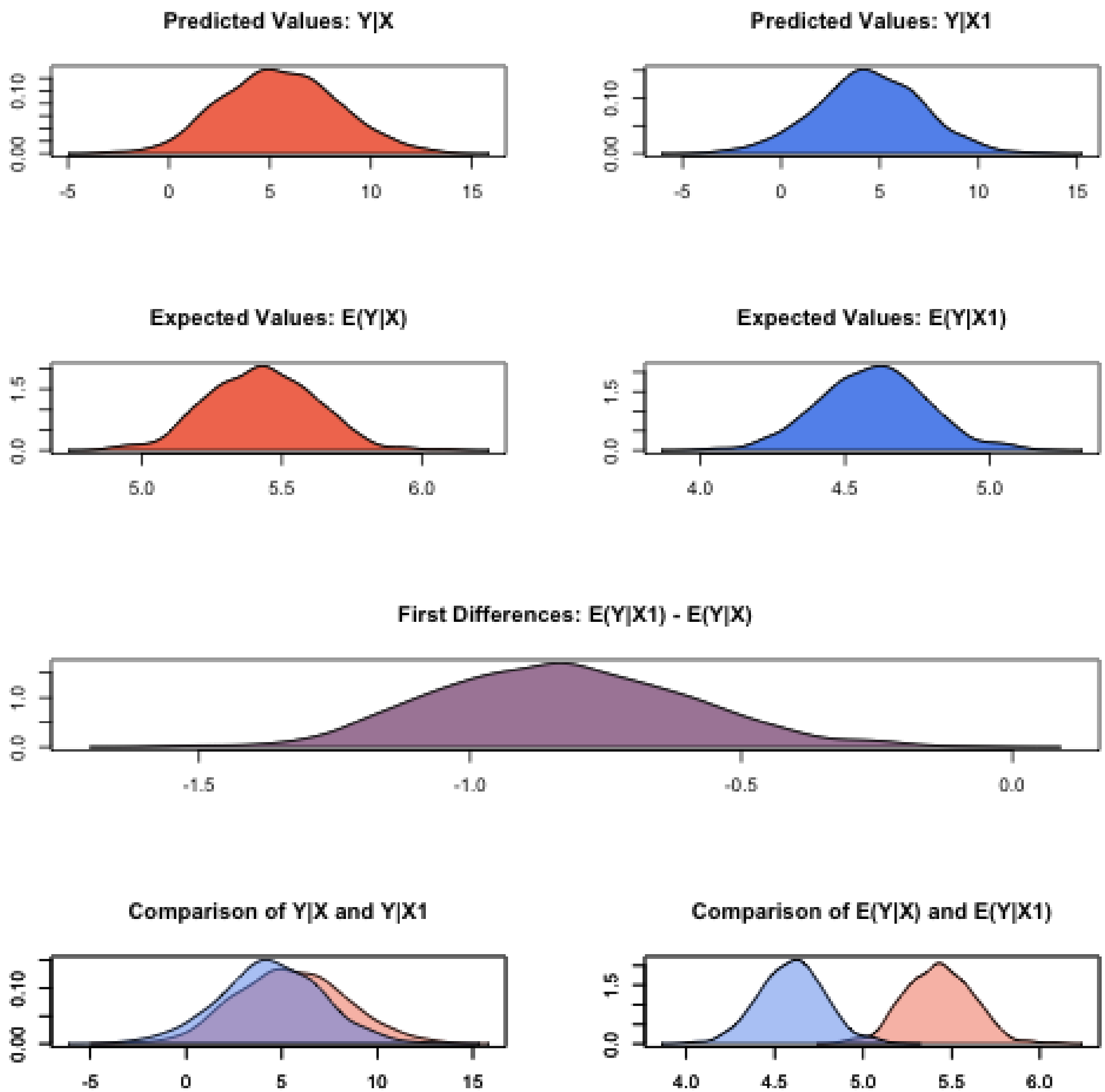


Figure 2.20: Zelig-probit

- The predicted value (qi$pr) is a draw from a Bernoulli distribution with mean π_i.
- The first difference (qi$fd) in expected values is defined as

$$\text{FD} = \Pr(Y = 1 \mid x_1) - \Pr(Y = 1 \mid x).$$

- The risk ratio (qi$rr) is defined as

$$\text{RR} = \Pr(Y = 1 \mid x_1) / \Pr(Y = 1 \mid x).$$

- In conditional prediction models, the average expected treatment effect (att.ev) for the treatment group is

$$\frac{1}{\sum_{i=1}^n t_i} \sum_{i:t_i=1}^n \{Y_i(t_i = 1) - E[Y_i(t_i = 0)]\},$$

  where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. Variation in the simulations are due to uncertainty in simulating $E[Y_i(t_i = 0)]$, the counterfactual expected value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.
- In conditional prediction models, the average predicted treatment effect (att.pr) for the treatment group is

$$\frac{1}{\sum_{i=1}^n t_i} \sum_{i:t_i=1}^n \left\{Y_i(t_i = 1) - Y_i(\widehat{t_i = 0})\right\},$$

  where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. Variation in the simulations are due to uncertainty in simulating $Y_i(\widehat{t_i = 0})$, the counterfactual predicted value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.

Output Values

The output of each Zelig command contains useful information which you may view. For example, if you run `z.out <- zelig(y ~ x, model = probit, data)`, then you may examine the available information in `z.out` by using `names(z.out)`, see the coefficients by using z.out$coefficients, and a default summary of information through `summary(z.out)`.

See also

The probit model is part of the stats package by . Advanced users may wish to refer to `help(glm)` and `help(family)`.

zelig-relogit

Rare Events Logistic Regression for Dichotomous Dependent Variables

The relogit procedure estimates the same model as standard logistic regression (appropriate when you have a dichotomous dependent variable and a set of explanatory variables; see), but the estimates are corrected for the bias that occurs when the sample is small or the observed events are rare (i.e., if the dependent variable has many more 1s than 0s or the reverse). The relogit procedure also optionally uses prior correction for case-control sampling designs.

Syntax

With reference classes:

```
z5 <- zrelogit$new()
z5$zelig(Y ~ X1 + X2, tau = NULL,
                       case.control = c("prior", "weighting"),
                       bias.correct = TRUE, robust = FALSE,
                       data = mydata, ...)
z5$setx()
z5$sim()
```

With the Zelig 4 compatibility wrappers:

```
z.out <- zelig(Y ~ X1 + X2, model = "relogit", tau = NULL,
                       case.control = c("prior", "weighting"),
                       bias.correct = TRUE, robust = FALSE,
                       data = mydata, ...)
x.out <- setx(z.out)
s.out <- sim(z.out, x = x.out)
```

Arguments

The relogit procedure supports four optional arguments in addition to the standard arguments for zelig(). You may additionally use:

- tau: a vector containing either one or two values for τ, the true population fraction of ones. Use, for example, tau = c(0.05, 0.1) to specify that the lower bound on tau is 0.05 and the upper bound is 0.1. If left unspecified, only finite-sample bias correction is performed, not case-control correction.
- case.control: if tau is specified, choose a method to correct for case-control sampling design: "prior" (default) or "weighting".
- bias.correct: a logical value of TRUE (default) or FALSE indicating whether the intercept should be corrected for finite sample (rare events) bias.

Note that if tau = NULL, bias.correct = FALSE, the relogit procedure performs a standard logistic regression without any correction.

Example 1: One Tau with Prior Correction and Bias Correction

Due to memory and space considerations, the data used here are a sample drawn from the full data set used in King and Zeng, 2001, The proportion of militarized interstate conflicts to the absence of disputes is $\tau = 1,042/303,772 \approx 0.00343$. To estimate the model,

```
data(mid)
```

```
z.out1 <- zelig(conflict ~ major + contig + power + maxdem + mindem + years, data = mid, model = "re
```

```
## How to cite this model in Zelig:
##   Kosuke Imai, Gary King, and Olivia Lau. 2014.
##   relogit: Rare Events Logistic Regression for Dichotomous Dependent Variables
##   in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
##   http://datascience.iq.harvard.edu/zelig
```

Summarize the model output:

```
summary(z.out1)
```

```
## Model: 1
## Call:  relogit(formula = cbind(conflict, 1 - conflict) ~ major + contig +
##     power + maxdem + mindem + years, data = ., tau = 0.00343020423212146,
##     bias.correct = TRUE, case.control = "prior")
##
## Coefficients:
## (Intercept)        major       contig        power       maxdem
##     -7.5084       2.4320       4.1080       1.0536       0.0480
##      mindem        years
##     -0.0641      -0.0629
##
## Degrees of Freedom: 3125 Total (i.e. Null);  3119 Residual
## Null Deviance:       3980
## Residual Deviance: 1870  AIC: 1880
## Next step: Use 'setx' method
```

Set the explanatory variables to their means:

```
x.out1 <- setx(z.out1)
```

Simulate quantities of interest:

```
s.out1 <- sim(z.out1, x = x.out1)
summary(s.out1)
```

```
##
##  sim x :
##  -----
## ev
##          mean       sd      50%     2.5%    97.5%
## [1,] 0.002393 0.000155 0.002388 0.002103 0.002708
## pv
##          0     1
## [1,] 0.998 0.002
```

```
plot(s.out1)
```

Example 2: One Tau with Weighting, Robust Standard Errors, and Bias Correction

Suppose that we wish to perform case control correction using weighting (rather than the default prior correction). To estimate the model:

```
z.out2 <- zelig(conflict ~ major + contig + power + maxdem + mindem + years, data = mid, model = "rel
```

```
## Error: unused argument (robust = TRUE)
```

Summarize the model output:

```
summary(z.out2)
```

```
## Model: 1
## Call:
## stats::lm(formula = unem ~ gdp + trade + capmob + as.factor(country),
##     data = .)
##
```

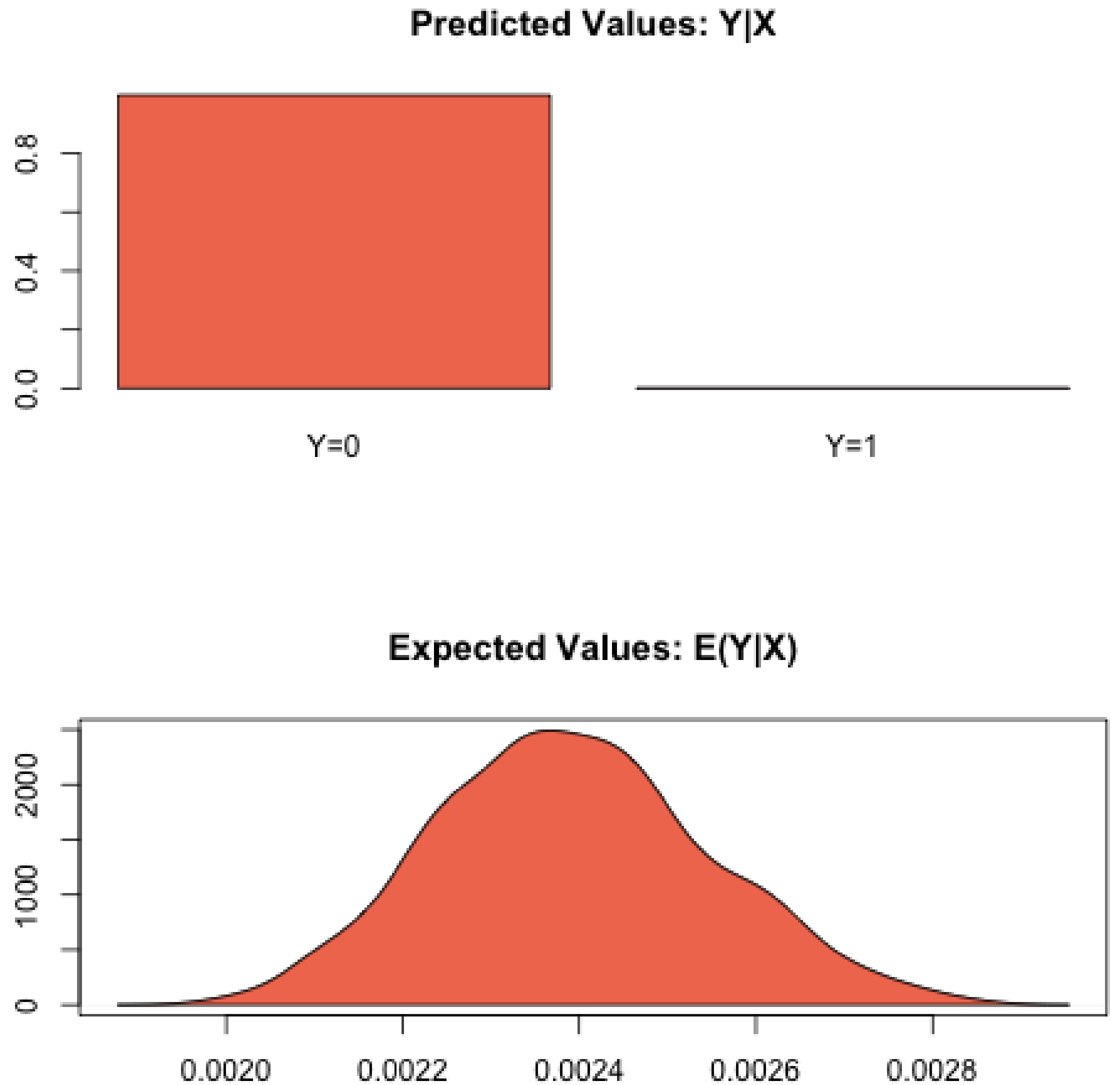


Figure 2.21: Zelig-relogit

```
## Coefficients:
##                     (Intercept)                           gdp
##                          -5.843                        -0.110
##                           trade                        capmob
##                           0.144                         0.815
##        as.factor(country)Belgium      as.factor(country)Canada
##                          -1.599                         6.759
##        as.factor(country)Denmark     as.factor(country)Finland
##                           4.311                         4.810
##         as.factor(country)France       as.factor(country)Italy
##                           6.905                         9.290
##          as.factor(country)Japan as.factor(country)Netherlands
##                           5.459                        -1.459
##         as.factor(country)Norway      as.factor(country)Sweden
##                          -2.754                         0.925
## as.factor(country)United Kingdom  as.factor(country)United States
##                           5.601                        10.066
##   as.factor(country)West Germany
##                           3.364
##
## Next step: Use 'setx' method
```

Set the explanatory variables to their means:

```
x.out2 <- setx(z.out2)
```

Simulate quantities of interest:

```
s.out2 <- sim(z.out2, x = x.out2)
summary(s.out2)
```

```
##
##  sim x :
##  -----
## ev
##    mean     sd  50%  2.5% 97.5%
## 1 10.59 0.4011 10.6 9.818  11.4
## pv
##    mean     sd  50%  2.5% 97.5%
## 1 10.59 0.4011 10.6 9.818  11.4
```

Example 3: Two Taus with Bias Correction and Prior Correction

Suppose that we did not know that $\tau \approx 0.00343$, but only that it was somewhere between $(0.002, 0.005)$. To estimate a model with a range of feasible estimates for τ (using the default prior correction method for case control correction):

```
z.out2 <- zelig(conflict ~ major + contig + power + maxdem + mindem + years, data = mid, model = "rel
```

```
## How to cite this model in Zelig:
##   Kosuke Imai, Gary King, and Olivia Lau. 2014.
##   relogit: Rare Events Logistic Regression for Dichotomous Dependent Variables
##   in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
##   http://datascience.iq.harvard.edu/zelig
```

Summarize the model output:

```
z.out2
```

```
## Model: 1$lower.estimate
##
## Call:  (function (formula, data = sys.parent(), tau = NULL, bias.correct = TRUE,
##     case.control = "prior", ...)
## {
##     mf <- match.call()
##     mf$tau <- mf$bias.correct <- mf$case.control <- NULL
##     if (!is.null(tau)) {
##         tau <- unique(tau)
##         if (length(case.control) > 1)
##             stop("You can only choose one option for case control correction.")
##         ck1 <- grep("p", case.control)
##         ck2 <- grep("w", case.control)
##         if (length(ck1) == 0 & length(ck2) == 0)
##             stop("choose either case.control = \"prior\" ", "or case.control = \"weighting\"")
##         if (length(ck2) == 0)
##             weighting <- FALSE
##         else weighting <- TRUE
##     }
##     else weighting <- FALSE
##     if (length(tau) > 2)
##         stop("tau must be a vector of length less than or equal to 2")
##     else if (length(tau) == 2) {
##         mf[[1]] <- relogit
##         res <- list()
##         mf$tau <- min(tau)
##         res$lower.estimate <- eval(as.call(mf), parent.frame())
##         mf$tau <- max(tau)
##         res$upper.estimate <- eval(as.call(mf), parent.frame())
##         res$formula <- formula
##         class(res) <- c("Relogit2", "Relogit")
##         return(res)
##     }
##     else {
##         mf[[1]] <- glm
##         mf$family <- binomial(link = "logit")
##         y2 <- model.response(model.frame(mf$formula, data))
##         if (is.matrix(y2))
##             y <- y2[, 1]
##         else y <- y2
##         ybar <- mean(y)
##         if (weighting) {
##             w1 <- tau/ybar
##             w0 <- (1 - tau)/(1 - ybar)
##             wi <- w1 * y + w0 * (1 - y)
##             mf$weights <- wi
##         }
##         res <- eval(as.call(mf), parent.frame())
##         res$call <- match.call(expand.dots = TRUE)
##         res$tau <- tau
##         X <- model.matrix(res)
##         if (bias.correct) {
##             pihat <- fitted(res)
##             if (is.null(tau))
##                 wi <- rep(1, length(y))
##             else if (weighting)
```

```
##                 res$weighting <- TRUE
##             else {
##                 w1 <- tau/ybar
##                 w0 <- (1 - tau)/(1 - ybar)
##                 wi <- w1 * y + w0 * (1 - y)
##                 res$weighting <- FALSE
##             }
##             W <- pihat * (1 - pihat) * wi
##             Qdiag <- lm.influence(lm(y ~ X - 1, weights = W))$hat/W
##             if (is.null(tau))
##                 xi <- 0.5 * Qdiag * (2 * pihat - 1)
##             else xi <- 0.5 * Qdiag * ((1 + w0) * pihat - w0)
##             res$coefficients <- res$coefficients - lm(xi ~ X -
##                 1, weights = W)$coefficients
##             res$bias.correct <- TRUE
##         }
##         else res$bias.correct <- FALSE
##         if (!is.null(tau) & !weighting) {
##             if (tau <= 0 || tau >= 1)
##                 stop("\ntau needs to be between 0 and 1.\n")
##             res$coefficients["(Intercept)"] <- res$coefficients["(Intercept)"] -
##                 log(((1 - tau)/tau) * (ybar/(1 - ybar)))
##             res$prior.correct <- TRUE
##             res$weighting <- FALSE
##         }
##         else res$prior.correct <- FALSE
##         if (is.null(res$weighting))
##             res$weighting <- FALSE
##         res$linear.predictors <- t(res$coefficients) %*% t(X)
##         res$fitted.values <- 1/(1 + exp(-res$linear.predictors))
##         res$zelig <- "Relogit"
##         class(res) <- c("Relogit", "glm")
##         return(res)
##     }
## })(formula = cbind(conflict, 1 - conflict) ~ major + contig +
##     power + maxdem + mindem + years, data = ., tau = 0.002)
##
## Coefficients:
## (Intercept)        major       contig        power       maxdem
##     -8.0492       2.4320       4.1079       1.0536       0.0480
##      mindem        years
##     -0.0641      -0.0629
##
## Degrees of Freedom: 3125 Total (i.e. Null);  3119 Residual
## Null Deviance:       3980
## Residual Deviance: 1870  AIC: 1880
##
## $upper.estimate
##
## Call:  (function (formula, data = sys.parent(), tau = NULL, bias.correct = TRUE,
##     case.control = "prior", ...)
## {
##     mf <- match.call()
##     mf$tau <- mf$bias.correct <- mf$case.control <- NULL
##     if (!is.null(tau)) {
##         tau <- unique(tau)
##         if (length(case.control) > 1)
##             stop("You can only choose one option for case control correction.")
```

```
##         ck1 <- grep("p", case.control)
##         ck2 <- grep("w", case.control)
##         if (length(ck1) == 0 & length(ck2) == 0)
##             stop("choose either case.control = \"prior\" ", "or case.control = \"weighting\"")
##         if (length(ck2) == 0)
##             weighting <- FALSE
##         else weighting <- TRUE
##     }
##     else weighting <- FALSE
##     if (length(tau) > 2)
##         stop("tau must be a vector of length less than or equal to 2")
##     else if (length(tau) == 2) {
##         mf[[1]] <- relogit
##         res <- list()
##         mf$tau <- min(tau)
##         res$lower.estimate <- eval(as.call(mf), parent.frame())
##         mf$tau <- max(tau)
##         res$upper.estimate <- eval(as.call(mf), parent.frame())
##         res$formula <- formula
##         class(res) <- c("Relogit2", "Relogit")
##         return(res)
##     }
##     else {
##         mf[[1]] <- glm
##         mf$family <- binomial(link = "logit")
##         y2 <- model.response(model.frame(mf$formula, data))
##         if (is.matrix(y2))
##             y <- y2[, 1]
##         else y <- y2
##         ybar <- mean(y)
##         if (weighting) {
##             w1 <- tau/ybar
##             w0 <- (1 - tau)/(1 - ybar)
##             wi <- w1 * y + w0 * (1 - y)
##             mf$weights <- wi
##         }
##         res <- eval(as.call(mf), parent.frame())
##         res$call <- match.call(expand.dots = TRUE)
##         res$tau <- tau
##         X <- model.matrix(res)
##         if (bias.correct) {
##             pihat <- fitted(res)
##             if (is.null(tau))
##                 wi <- rep(1, length(y))
##             else if (weighting)
##                 res$weighting <- TRUE
##             else {
##                 w1 <- tau/ybar
##                 w0 <- (1 - tau)/(1 - ybar)
##                 wi <- w1 * y + w0 * (1 - y)
##                 res$weighting <- FALSE
##             }
##             W <- pihat * (1 - pihat) * wi
##             Qdiag <- lm.influence(lm(y ~ X - 1, weights = W))$hat/W
##             if (is.null(tau))
##                 xi <- 0.5 * Qdiag * (2 * pihat - 1)
##             else xi <- 0.5 * Qdiag * ((1 + w0) * pihat - w0)
##             res$coefficients <- res$coefficients - lm(xi ~ X -
```

```
##                 1, weights = W)$coefficients
##             res$bias.correct <- TRUE
##         }
##         else res$bias.correct <- FALSE
##         if (!is.null(tau) & !weighting) {
##             if (tau <= 0 || tau >= 1)
##                 stop("\ntau needs to be between 0 and 1.\n")
##             res$coefficients["(Intercept)"] <- res$coefficients["(Intercept)"] -
##                 log(((1 - tau)/tau) * (ybar/(1 - ybar)))
##             res$prior.correct <- TRUE
##             res$weighting <- FALSE
##         }
##         else res$prior.correct <- FALSE
##         if (is.null(res$weighting))
##             res$weighting <- FALSE
##         res$linear.predictors <- t(res$coefficients) %*% t(X)
##         res$fitted.values <- 1/(1 + exp(-res$linear.predictors))
##         res$zelig <- "Relogit"
##         class(res) <- c("Relogit", "glm")
##         return(res)
##     }
## })(formula = cbind(conflict, 1 - conflict) ~ major + contig +
##     power + maxdem + mindem + years, data = ., tau = 0.005)
##
## Coefficients:
## (Intercept)        major       contig        power       maxdem
##     -7.1300       2.4320       4.1080       1.0536       0.0480
##      mindem        years
##     -0.0641      -0.0629
##
## Degrees of Freedom: 3125 Total (i.e. Null);  3119 Residual
## Null Deviance:       3980
## Residual Deviance: 1870  AIC: 1880
##
## $formula
## cbind(conflict, 1 - conflict) ~ major + contig + power + maxdem +
##     mindem + years
## <environment: 0x7f8eccf6a9c8>
##
## attr(,"class")
## [1] "Relogit2" "Relogit"
## Next step: Use 'setx' method
```

Set the explanatory variables to their means:

```
x.out2 <- setx(z.out2)
```

Simulate quantities of interest:

```
s.out <- sim(z.out2, x = x.out2)
```

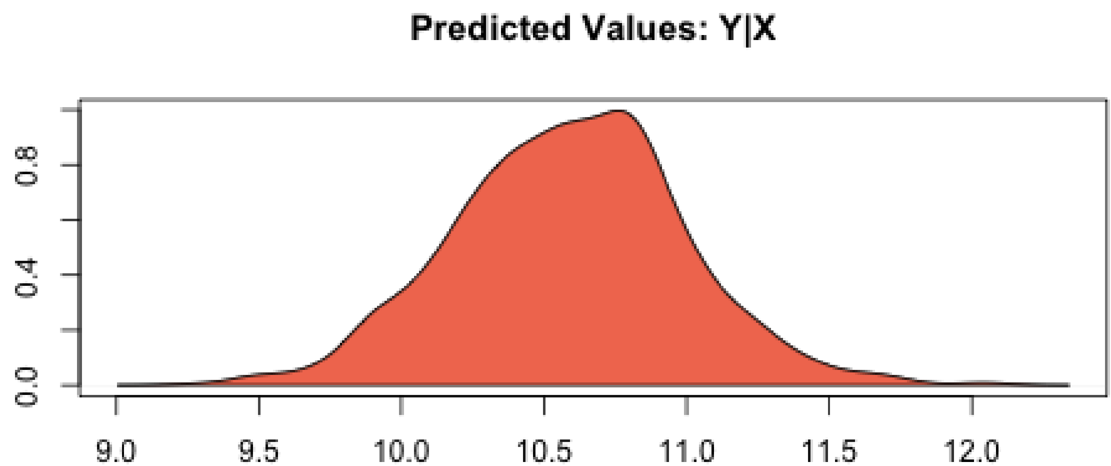
```
## Error: no applicable method for 'vcov' applied to an object of class
## "c('Relogit2', 'Relogit')"
```

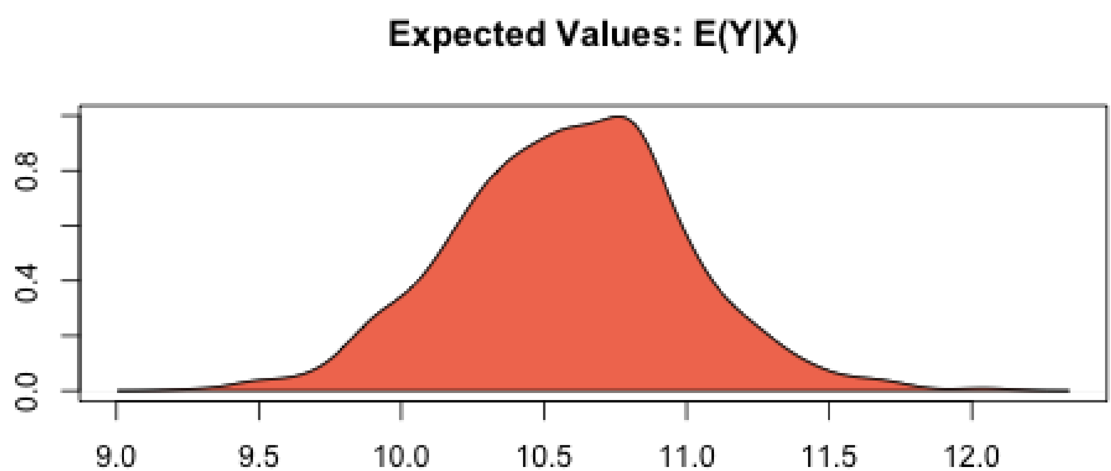
```
summary(s.out2)
```

```
##
##  sim x :
##  -----
```

```
## ev
##    mean     sd  50%  2.5% 97.5%
## 1 10.59 0.4011 10.6 9.818  11.4
## pv
##    mean     sd  50%  2.5% 97.5%
## 1 10.59 0.4011 10.6 9.818  11.4
```

```
plot(s.out2)
```





The cost of giving a range of values for τ is that point estimates are not available for quantities of interest. Instead, quantities are presented as confidence intervals with significance less than or equal to a specified level (e.g., at least 95% of the simulations are contained in the nominal 95% confidence interval).

Model

- Like the standard logistic regression, the *stochastic component* for the rare events logistic regression is:

$$Y_i \;\sim\; \text{Bernoulli}(\pi_i),$$

  where Y_i is the binary dependent variable, and takes a value of either 0 or 1.

- The *systematic component* is:

$$\pi_i \;=\; \frac{1}{1+\exp(-x_i\beta)}.$$

- If the sample is generated via a case-control (or choice-based) design, such as when drawing all events (or "cases") and a sample from the non-events (or "controls") and going backwards to collect the explanatory variables, you must correct for selecting on the dependent variable. While the slope coefficients are approximately unbiased, the constant term may be significantly biased. Zelig has two methods for case control correction:

  1. The "prior correction" method adjusts the intercept term. Let τ be the true population fraction of events, $\bar{y}$ the fraction of events in the sample, and $\hat{\beta}_0$ the uncorrected intercept term. The corrected intercept β_0 is:

$$\beta = \hat{\beta}_0 - \ln\left[\left(\frac{1-\tau}{\tau}\right)\left(\frac{\bar{y}}{1-\bar{y}}\right)\right].$$

  2. The "weighting" method performs a weighted logistic regression to correct for a case-control sampling design. Let the 1 subscript denote observations for which the dependent variable is observed as a 1, and the 0 subscript denote observations for which the dependent variable is observed as a 0. Then the vector of weights w_i

$$\begin{aligned} w_1 =& \quad \frac{\tau}{\bar{y}} \\ w_0 =& \quad \frac{(1-\tau)}{(1-\bar{y})} \\ w_i =& \quad w_1 Y_i + w_0(1-Y_i) \end{aligned}$$

  If τ is unknown, you may alternatively specify an upper and lower bound for the possible range of τ. In this case, the relogit procedure uses "robust Bayesian" methods to generate a confidence interval (rather than a point estimate) for each quantity of interest. The nominal coverage of the confidence interval is at least as great as the actual coverage.

- By default, estimates of the the coefficients β are bias-corrected to account for finite sample or rare events bias. In addition, quantities of interest, such as predicted probabilities, are also corrected of rare-events bias. If $\widehat{\beta}$ are the uncorrected logit coefficients and bias($\widehat{\beta}$) is the bias term, the corrected coefficients $\tilde{\beta}$ are

$$\widehat{\beta} - \text{bias}(\widehat{\beta}) = \tilde{\beta}$$

  The bias term is

$$\text{bias}(\widehat{\beta}) = (X'WX)^{-1}X'W\xi$$

  where

$$\begin{aligned} \xi_i =& \quad 0.5Q_{ii}\Big((1+w-1)\widehat{\pi}_i - w_1\Big) \\ Q =& \quad X(X'WX)^{-1}X' \end{aligned}$$
$$W = \text{diag}\{\widehat{\pi}_i(1-\widehat{\pi}_i)w_i\}$$

  where w_i and w_1 are given in the "weighting" section above.

Quantities of Interest

- For either one or no τ:
  - The expected values (qi$ev) for the rare events logit are simulations of the predicted probability
    $$E(Y) = \pi_i = \frac{1}{1 + \exp(-x_i \beta)},$$
    given draws of β from its posterior.
  - The predicted value (qi$pr) is a draw from a binomial distribution with mean equal to the simulated π_i.
  - The first difference (qi$fd) is defined as
    $$\text{FD} = \Pr(Y = 1 \mid x_1, \tau) - \Pr(Y = 1 \mid x, \tau).$$
  - The risk ratio (qi$rr) is defined as
    $$\text{RR} = \Pr(Y = 1 \mid x_1, \tau) \ / \ \Pr(Y = 1 \mid x, \tau).$$
- For a range of τ defined by $[\tau_1, \tau_2]$, each of the quantities of interest are $n \times 2$ matrices, which report the lower and upper bounds, respectively, for a confidence interval with nominal coverage at least as great as the actual coverage. At worst, these bounds are conservative estimates for the likely range for each quantity of interest. Please refer to for the specific method of calculating bounded quantities of interest.
- In conditional prediction models, the average expected treatment effect (att.ev) for the treatment group is
  $$\frac{1}{\sum_{i=1}^n t_i} \sum_{i:t_i=1}^n \left\{ Y_i(t_i = 1) - E[Y_i(t_i = 0)] \right\},$$
  where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. Variation in the simulations are due to uncertainty in simulating $E[Y_i(t_i = 0)]$, the counterfactual expected value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.
- In conditional prediction models, the average predicted treatment effect (att.pr) for the treatment group is
  $$\frac{1}{\sum_{i=1}^n t_i} \sum_{i:t_i=1}^n \left\{ Y_i(t_i = 1) - Y_i(\widehat{t_i = 0}) \right\},$$
  where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups. Variation in the simulations are due to uncertainty in simulating $Y_i(\widehat{t_i = 0})$, the counterfactual predicted value of Y_i for observations in the treatment group, under the assumption that everything stays the same except that the treatment indicator is switched to $t_i = 0$.

Output Values

The output of each Zelig command contains useful information which you may view. For example, if you run `z.out <- zelig(y ~ x, model = relogit, data)`, then you may examine the available information in `z.out` by using `names(z.out)`, see the coefficients by using z.out$coefficients, and a default summary of information through `summary(z.out)`.

Differences with Stata Version

The Stata version of ReLogit and the R implementation differ slightly in their coefficient estimates due to differences in the matrix inversion routines implemented in R and Stata. Zelig uses orthogonal-triangular decomposition (through lm.influence()) to compute the bias term, which is more numerically stable than standard matrix calculations.

See also

zelig-tobit

Linear Regression for a Left-Censored Dependent Variable

Tobit regression estimates a linear regression model for a left-censored dependent variable, where the dependent variable is censored from below. While the classical tobit model has values censored at 0, you may select another censoring point. For other linear regression models with fully observed dependent variables, see Bayesian regression (), maximum likelihood normal regression (), or least squares ().

Syntax

```
z5 <- ztobit$new()
z5$zelig(Y ~ X1 + X2, below = 0, above = Inf, data = mydata)
z5$setx()
z5$sim()
```

With the Zelig 4 compatibility wrappers:

```
z.out <- zelig(Y ~ X1 + X2, below = 0, above = Inf, model = "tobit", data = mydata)
x.out <- setx(z.out)
s.out <- sim(z.out, x = x.out)
```

Inputs

zelig() accepts the following arguments to specify how the dependent variable is censored.

- `below`: (defaults to 0) The point at which the dependent variable is censored from below. If any values in the dependent variable are observed to be less than the censoring point, it is assumed that that particular observation is censored from below at the observed value. (See for a Bayesian implementation that supports both left and right censoring.)
- robust: defaults to FALSE. If TRUE, zelig() computes robust standard errors based on sandwich estimators (see and) and the options selected in cluster.
- cluster: if robust = TRUE, you may select a variable to define groups of correlated observations. Let x3 be a variable that consists of either discrete numeric values, character strings, or factors that define strata. Then

  ```
  > z.out <- zelig(y ~ x1 + x2, robust = TRUE, cluster = "x3",
                   model = "tobit", data = mydata)
  ```

  means that the observations can be correlated within the strata defined by the variable x3, and that robust standard errors should be calculated according to those clusters. If robust = TRUE but cluster is not specified, zelig() assumes that each observation falls into its own cluster.

Zelig users may wish to refer to `help(survreg)` for more information.

Examples

Basic Example Attaching the sample dataset:

```
data(tobin)
```

Estimating linear regression using `tobit`:

```
z.out <- zelig(durable ~ age + quant, model = "tobit", data = tobin)

## How to cite this model in Zelig:
##   Kosuke Imai, Gary King, Olivia Lau. 2011.
##   tobit: Linear regression for Left-Censored Dependent Variable
##   in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
##   http://datascience.iq.harvard.edu/zelig
```

Setting values for the explanatory variables to their sample averages:

```
x.out <- setx(z.out)
```

Simulating quantities of interest from the posterior distribution given `x.out`.

```
s.out1 <- sim(z.out, x = x.out)
```

```
summary(s.out1)

##
##  sim x :
##  -----
## ev
##   mean     sd   50%   2.5% 97.5%
## 1 1.55 0.6187 1.476 0.5527 2.965
## pv
##      mean    sd  50% 2.5% 97.5%
## [1,] 3.23 4.082 1.81    0  13.8
```

Simulating First Differences Set explanatory variables to their default(mean/mode) values, with high (80th percentile) and low (20th percentile) liquidity ratio (`quant`):

```
x.high <- setx(z.out, quant = quantile(tobin$quant, prob = 0.8))
x.low <- setx(z.out, quant = quantile(tobin$quant, prob = 0.2))
```

Estimating the first difference for the effect of high versus low liquidity ratio on duration(`durable`):

```
s.out2 <- sim(z.out, x = x.high, x1 = x.low)
```

```
summary(s.out2)

##
##  sim x :
##  -----
## ev
##   mean     sd   50%   2.5% 97.5%
## 1 1.18 0.7514 1.041 0.1329 2.995
## pv
##       mean    sd   50% 2.5% 97.5%
## [1,] 2.884 3.884 1.182    0 13.02
##
##  sim x1 :
##  -----
## ev
##   mean     sd   50%   2.5% 97.5%
## 1 2.06 0.9851 1.911 0.6058 4.218
## pv
##       mean    sd  50% 2.5% 97.5%
## [1,] 3.495 4.289 1.98    0 14.66
```

```
## fd
##      mean    sd    50%   2.5% 97.5%
## 1 0.8797 1.184 0.8411 -1.315 3.409
```

```
plot(s.out1)
```

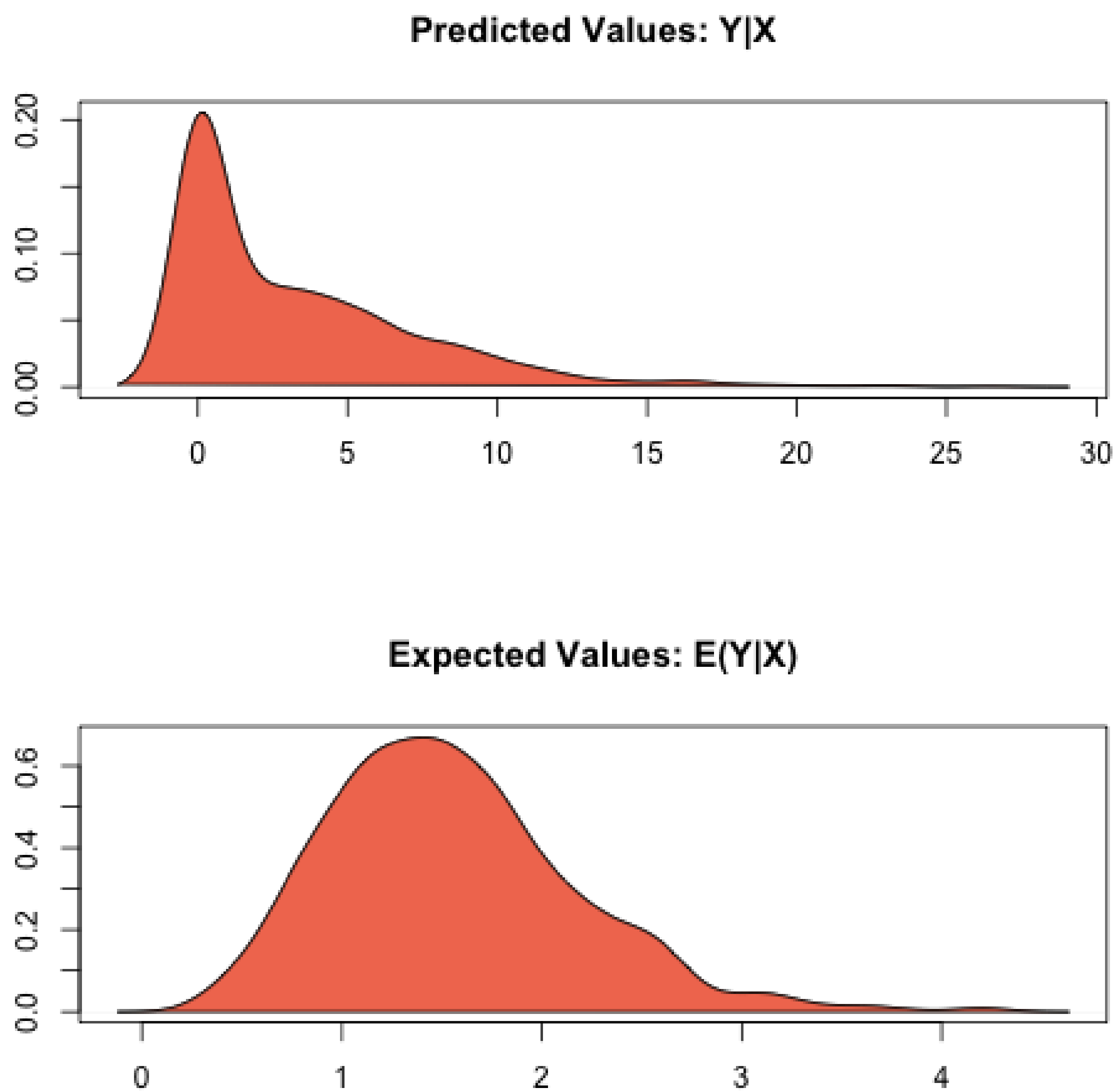


Figure 2.22: Zelig-tobit

Model

- Let Y_i^* be a latent dependent variable which is distributed with *stochastic* component

$$Y_i^* \sim \quad \text{Normal}(\mu_i, \sigma^2)$$

where μ_i is a vector means and σ^2 is a scalar variance parameter. Y_i^* is not directly observed, however. Rather we observed Y_i which is defined as:

$$Y_i = \begin{cases} Y_i^* & \text{if} \quad c < Y_i^* \\ c & \text{if} \quad c \ge Y_i^* \end{cases}$$

where c is the lower bound below which Y_i^* is censored.

- The *systematic component* is given by

$$\mu_i = \quad x_i \beta,$$

where x_i is the vector of k explanatory variables for observation i and β is the vector of coefficients.

Quantities of Interest

- The expected values (`qi$ev`) for the tobit regression model are the same as the expected value of $Y*$:

$$E(Y^*|X) = \mu_i = x_i \beta$$

- The first difference (`qi$fd`) for the tobit regression model is defined as

$$\text{FD} = E(Y^* \mid x_1) - E(Y^* \mid x).$$

- In conditional prediction models, the average expected treatment effect (`qi$att.ev`) for the treatment group is

$$\frac{1}{\sum t_i} \sum_{i:t_i=1} [E[Y_i^*(t_i = 1)] - E[Y_i^*(t_i = 0)]],$$

where t_i is a binary explanatory variable defining the treatment ($t_i = 1$) and control ($t_i = 0$) groups.

Output Values

The output of each Zelig command contains useful information which you may view. For example, if you run:

```
z.out <- zelig(y ~ x, model = "tobit", data)
```

then you may examine the available information in ‘‘z.out‘.

See also

The tobit function is part of the survival library by Terry Therneau, ported to R by Thomas Lumley. Advanced users may wish to refer to `help(survfit)` in the survival library.

CHAPTER THREE

FREQUENTLY ASKED QUESTIONS

If you find a bug, or cannot figure something out after reading through the FAQs below, please send your question to the Zelig listserv at: https://groups.google.com/forum/#!forum/zelig-statistical-software. Please explain exactly what you did and include the full error message, including the traceback(). You should get an answer from the developers or another user in short order.

3.1 Why can't I install Zelig?

We recommend that you first check your internet connection, as you must be connected to install packages. In addition, there are a few platform-specific reasons why you may be having installation problems:

- **On Windows**: If you are using the very latest version of R, you may not be able to install Zelig until we update Zelig to work with this latest release. Currently Zelig 5.0-1 is compatible with R (>= 3.0.2). If you wish to install Zelig in the interim, install the appropriate version of R and try to reinstall Zelig.
- **On Mac or Linux systems**: If you get the following warning message at the end of your installation:

  ```
  > Installation of package VGAM had non-zero exit status in ...
  ```

  this means that you were not able to install VGAM properly. Make sure that you have the g77 Fortran compiler. For Intel Macs, download the Apple developer tools. After installation, try to install Zelig again.

If neither solution works, feel free email the Zelig mailing list directly at: https://groups.google.com/forum/#!forum/zelig-statistical-software.

3.2 Why can't I install R?

If you have problems installing R, you should search the internet for the R help mailing list, check out technical Q & A forums (e.g., StackOverflow), or email the Zelig mailing list directly at: https://groups.google.com/forum/#!forum/zelig-statistical-software.

3.3 Why can't I load data?

It is likely that the reason you are unable to load data because you have not specified the correct working directory (e.g., the location of the data you are trying to load). You should specify you working directory use the `setwd()` function in which you will include the the file path to your working director. For example, if I wanted to load a file that is my *Documents* folder, I must first:

```
> setwd("path/to/Documents")
```

File paths can be found by right clicking the working directory folder in any file browser and clicking "Get Info" (on Mac) or "Properties" (on Windows). Black-slashes (\) in file paths copied from the "Properties" link on Windows machines must be replace with forward-slashes (/). For example, the Windows path: `C:\Program Files\R`, would be typed as `C:/Program Files/R`.

3.4 R is neat. How can I find out more?

R is a collective project with contributors from all over the world. Their website (http://www.r-project.org) has more information on the R project, R packages, conferences, and other learning material.

CHAPTER

FOUR

ABOUT ZELIG

Zelig is an open-source project developed and maintained by the Data Science group at Harvard's Institute for Quantitative Social Science. It was originally conceived and created by Kosuke Imai, Gary King, and Olivia Lau in 2007. The name is borrowed from Woody Allen's movie with the same name, Zelig. Leonard Zelig is a fictional character who takes on the characteristics of any strong personality around. Likewise, the Zelig statistical software easily adapts to any statistical model written in R, and in essence, takes the characteristics of any model.

Zelig leverages (R) code from many researchers and is designed to allow anyone to contribute their methods to it. Hence, we often refer to Zelig as "everyone's statistical software" and our aim is to make it, as well as the models it wraps, as accessible as possible. As such, it comes with self-contained documentation that minimizes startup costs, automates model summaries and graphics, and bridges existing R implementations through an intelligible call structure.

Contact: For questions, please join the Zelig mailing list: https://groups.google.com/forum/#!forum/zelig-statistical-software.

Original Authors:

- Gary King *(Principle Investigator)*
- Kosuke Imai
- Olivia Lau

The Zelig Team:

- James Honaker *(Project Lead)*
- Christine Choirat *(Lead Author)*
- Muhammed Y. Idris

4.1 Technical Vision

Zelig is a framework for interfacing a wide range of statistical models and analytic methods in a common and simple way. Above and beyond estimation, Zelig adds considerable infrastructure to existing heterogeneous R implementations by translating hard-to-interpret coefficients into quantities of interest (e.g., expected and predicted values) through a simple call structure. This includes many specific methods, based on likelihood, frequentist, Bayesian, robust Bayesian and nonparametric theories of inference. Developers are encouraged to add their R packages to the Zelig toolkit by writing a few simple bridge functions.

Additional features include:

- Dealing with missing data by combining multiply imputed datasets
- Automating statistical bootstrapping
- Improving parametric procedures by leveraging nonparametric matching methods
- Evaluating counterfactuals
- Allowing conditional population and super population inferences
- Automating the creation of replication data files

4.2 Release Notes

v 5.0-1

This release provides a set of core models, while simplifying the model wrapping process, and solving architectural problems by completely rewriting into R's Reference Classes for a fully object-oriented architecture.

Inheritance Tree