Generalized Linear Models - `glm`

Statistics 135

Autumn 2005

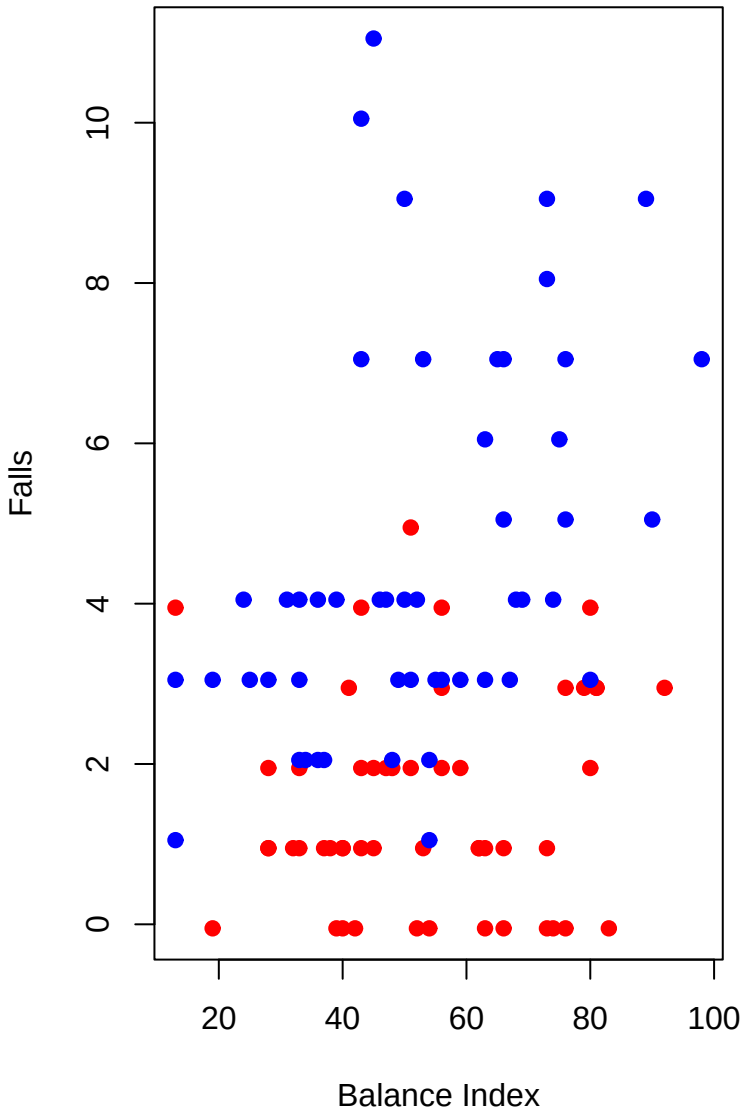
Fitting Generalized Linear Models - `glm`

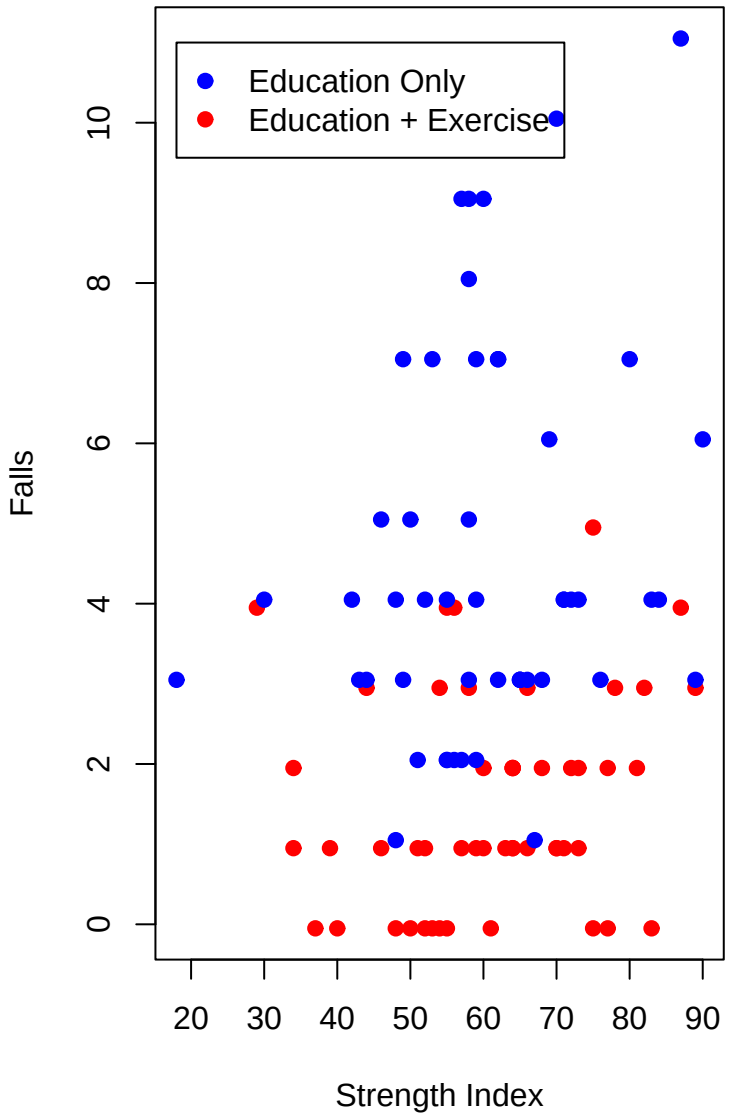
Example: Geriatric Study to Reduce Falls

100 subject were studied to investigate two treatments to which is better to reduce falls.

- y: number of falls during 6 months of study (self-reported)
- x_1: Treatment - 0 = education only, 1 = education + aerobic exercise
- x_2: Gender - 0 = female, 1 = male
- x_3: Balance Index (bigger is better)
- x_4: Strength Index (bigger is better)

The question of interest is how much does the treatment reduce the number of falls, after adjusting for Gender, Balance Index and Strength Index


Falls
0
2
4
6
8
10
20
40
60
80
100
Balance Index


Education Only
Education + Exercise
Falls
0
2
4
6
8
10
20
30
40
50
60
70
80
90
Strength Index

Lets analyze this data by Poisson regression with the log link, i.e. $y_i|X_i \sim P(\mu_i)$ where

$$\begin{aligned}
\log \mu_i &= \beta_0 + \beta_1 \text{Trt} + \beta_2 \text{Gender} + \beta_3 \text{Balance} + \beta_4 \text{Strength} \\
\mu_i &= \exp(\beta_0 + \beta_1 \text{Trt} + \beta_2 \text{Gender} + \beta_3 \text{Balance} + \beta_4 \text{Strength}) \\
&= \exp(\beta_0 + \beta_2 \text{Gender} + \beta_3 \text{Balance} + \beta_4 \text{Strength}) \times \exp(\beta_1 \text{Trt})
\end{aligned}$$

So for a fixed Gender, Balance, and Strength combination

$$\mu_i = \begin{cases} c & \text{Education only} \\ ce^{\beta_1} & \text{Education + Exercise} \end{cases}$$

where $c = \exp(\beta_0 + \beta_2 \text{Gender} + \beta_3 \text{Balance} + \beta_4 \text{Strength})$

This is an example of a multiplicative model. The effect of treatment on the mean is described by a multiplicative effect.

In addition to inference on β_1, we also want to perform inference on e^{β_1}.

We can analyze this model by the following code

```
> fall.glm <- glm(Falls ~ Treatment + Gender + Balance + Strength,
  family=poisson(), data=falls)
> summary(fall.glm)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.1854  -0.7819  -0.2564   0.5449   2.3625

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  0.489467   0.336869   1.453  0.14623
Treatment   -1.069403   0.133154  -8.031 9.64e-16 ***
Gender      -0.046606   0.119970  -0.388  0.69766
Balance      0.009470   0.002953   3.207  0.00134 **
Strength     0.008566   0.004312   1.986  0.04698 *
```

```
(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 199.19  on 99  degrees of freedom
Residual deviance: 108.79  on 95  degrees of freedom
AIC: 377.29

Number of Fisher Scoring iterations: 5
```

As mentioned last time, MLEs are approximately normally distributed. So one way of examining whether a β is significantly different from b_0 is by performing a z-test, which is of the form

$$z = \frac{\hat{\beta} - b_0}{se(\hat{\beta})}$$

and comparing this to a $N(0, 1)$.

The z-test for $H_0 : \beta_1 = 0$ has a $z = -8.031$ with a two-sided p-value $= 9.64 \times 10^{-16}$, which is pretty good evidence that exercise has an effect of the number of falls and its beneficial.

The base function for fitting GLIMs is `glm`

```
glm(formula, family = gaussian, data, weights, subset,
        na.action, start = NULL, etastart, mustart,
        offset, control = glm.control(...), model = TRUE,
        method = "glm.fit", x = FALSE, y = TRUE,
        contrasts = NULL, ...)

 formula: a symbolic description of the model to be fit. The
        details of model specification are given below.

  family: a description of the error distribution and link
        function to be used in the model. This can be a
        character string naming a family function, a family
        function or the result of a call to a family function.
        (See 'family' for details of family functions.)

 weights: an optional vector of weights to be used in the
        fitting process.
```

```
subset: an optional vector specifying a subset of observations
        to be used in the fitting process.

offset: this can be used to specify an _a priori_ known
        component to be included in the linear predictor
        during fitting.
```

The formula component works the same way as with the `lm` command.

There is one minor exception when dealing with binomial data. In that case, `y` needs to be a matrix with the first column giving the number of failures and the second column giving the number of successes.

Most functions that work for `lm` objects will also work for `glm` objects. For example we've seen that `summary` gives similar output. Another example is the `anova` function. However it works slightly differently.

```
> anova(fall.glm)
Analysis of Deviance Table

Model: poisson, link: log

Response: Falls

Terms added sequentially (first to last)


          Df Deviance Resid. Df Resid. Dev
NULL                         99    199.194
Treatment  1   75.209         98    123.985
Gender     1    0.625         97    123.360
Balance    1   10.594         96    112.766
Strength   1    3.976         95    108.790
```

We can still use the anova function to compare two models. For example,

```
> fallH0.glm <- update(fall.glm, . ~ . - Treatment)

> anova(fallH0.glm, fall.glm, test="Chisq")
Analysis of Deviance Table

Model 1: Falls ~ Gender + Balance + Strength
Model 2: Falls ~ Treatment + Gender + Balance + Strength
  Resid. Df Resid. Dev Df Deviance P(>|Chi|)
1        96     182.31
2        95     108.79  1    73.52 9.961e-18
```

This gives a different test that the z-test shown in the summary output. The test given in the ANOVA output is the likelihood ratio test, which uses a different distributional assumption than the z-test.

In the likelihood ratio test, the difference in the deviances (related to the log-likelihood) of the two nested models is approximately χ^2_{df} where df is the difference in the number of parameters of the two models.

Variances of Parameter Estimates

It may be necessary to get variances and covariances of parameter estimates in a GLIM. Uses may involve hypothesis tests, confidence intervals for β_i, or confidence intervals for $\mu = X\beta$ for a given set of predictors X.

This matrix can be gotten by the function `vcov`. Note that this function will also work with `lm` objects. An example of its use is to get pointwise confidence intervals for each β

```
> vcov(fall.glm)
              (Intercept)     Treatment        Gender       Balance
(Intercept)  0.1134809317 -4.449605e-04 -1.393692e-02 -4.949676e-04
Treatment   -0.0004449605  1.772996e-02 -2.577399e-03 -6.997795e-06
Gender      -0.0139369217 -2.577399e-03  1.439286e-02  1.628612e-07
Balance     -0.0004949676 -6.997795e-06  1.628612e-07  8.719750e-06
Strength    -0.0012143988 -4.030867e-05  1.268218e-04  1.007663e-07
                 Strength
(Intercept) -1.214399e-03
Treatment   -4.030867e-05
Gender       1.268218e-04
```

```
Balance      1.007663e-07
Strength     1.859437e-05
```

```
> se <- sqrt(diag(vcov(fall.glm)))
> ci <- coef(fall.glm) + qnorm(0.975) * cbind(se,-se)
```

```
> se
(Intercept)   Treatment      Gender     Balance    Strength
0.336869309 0.133153890 0.119970256 0.002952922 0.004312119
```

```
> ci
                                    se
(Intercept) -0.1707845489  1.14971888
Treatment   -1.3303793811 -0.80842572
Gender      -0.2817434434  0.18853132
Balance      0.0036823654  0.01525761
Strength     0.0001142306  0.01701743
```

As shown earlier, e^{β_1} is the multiplicative effect of adding exercise to the treatment. A confidence interval for this quantity can be calculated by exponentiating the interval for β_1.

```
> exp(coef(fall.glm)[2])
Treatment 0.3432135

> exp(ci[2,])
                  se
0.2643769 0.4455589
```

In this case, it appears that exercise lowers falls to about one third of the rate without it.

Changing the Link Function

Suppose that we wish to fit a different link function. For example model

$$\mu_i = \beta_0 + \beta_1 \text{Trt} + \beta_2 \text{Gender} + \beta_3 \text{Balance} + \beta_4 \text{Strength}$$

This can be done by modifying the `family` argument in the `glm` call

```
> fall.indent.glm <- glm(Falls ~ Treatment + Gender + Balance
    + Strength, family=poisson(link="identity"), data=falls)
Error: no valid set of coefficients has been found: please
supply starting values
In addition: Warning message:
NaNs produced in: log(x)
```

```
> summary(fall.indent.glm)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.2478  -0.7663  -0.2628   0.4651   2.4557

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  1.363700   0.270996   5.032 4.85e-07 ***
Treatment   -0.866151   0.101826  -8.506  < 2e-16 ***
Gender      -0.029844   0.103328  -0.289  0.77271
Balance      0.006935   0.002613   2.653  0.00797 **
Strength     0.006581   0.003590   1.833  0.06682 .

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 199.19  on 99  degrees of freedom
Residual deviance: 111.89  on 95  degrees of freedom
AIC: 380.39
```

```
Number of Fisher Scoring iterations: 6
```

For information about the possible distributions available and the link functions available for these distributions, see `help(family)`.

The following are common link function choices for different distributions

- Normal
  - Identity: $g(\mu) = \mu$
  - Log: $g(\mu) = \log \mu$
  - Inverse: $g(\mu) = \frac{1}{\mu}$
- Binomial
  - Logit: $g(\mu) = \log \frac{\mu}{1-\mu}$
  - Probit: $g(\mu) = \Phi^{-1}(\mu)$
  - Complementary Log-Log link: $g(\mu) = \log(-\log(\mu))$
  - Log: $g(\mu) = \log \mu$

- Poisson
  - Log: $g(\mu) = \log \mu$
  - Identity: $g(\mu) = \mu$
  - Square root: $g(\mu) = \sqrt{\mu}$
- Gamma
  - Inverse: $g(\mu) = \frac{1}{\mu}$
  - Log: $g(\mu) = \log \mu$
  - Identity: $g(\mu) = \mu$
- Inv-Normal
  - Inverse squared: $g(\mu) = \frac{1}{\mu^2}$
  - Inverse: $g(\mu) = \frac{1}{\mu}$
  - Log: $g(\mu) = \log \mu$
  - Identity: $g(\mu) = \mu$

Overdispersion

For Poisson and Binomial models, the dispersion parameter is fixed to be 1. However there can be data that look Poisson or Binomial but are more dispersed.

To examine this, we can use the `quasipoisson` or `quasibinomial` families in `glm`. When doing this, an overdispersion parameter is fit, which can also influence the fits of the βs.

```
> fall.quasi.glm <- glm(Falls ~ Treatment + Gender + Balance
    + Strength, family=quasipoisson(link="log"), data=falls)

> summary(fall.quasi.glm)

Call:
glm(formula = Falls ~ Treatment + Gender + Balance + Strength,
    family = quasipoisson(link = "log"), data = falls)
```

```
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.1854  -0.7819  -0.2564   0.5449   2.3625

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.489467   0.355076   1.378  0.17129
Treatment   -1.069403   0.140351  -7.620 1.89e-11 ***
Gender      -0.046606   0.126454  -0.369  0.71328
Balance      0.009470   0.003113   3.043  0.00303 **
Strength     0.008566   0.004545   1.885  0.06254 .

(Dispersion parameter for quasipoisson family taken to be 1.111017)

    Null deviance: 199.19  on 99  degrees of freedom
Residual deviance: 108.79  on 95  degrees of freedom
AIC: NA

Number of Fisher Scoring iterations: 5
```

There is another extension to this idea, which allows for more general relationships between the mean and the variance of the data. This can be done by setting the family to `quasi`. This is the analogue in regression to only making assumptions about the mean and variance (no normality assumption).

From the `family` help page

```
    link: (stuff about out families deleted)
          The 'quasi' family allows the links '"logit"',
          '"probit"', '"cloglog"',  '"identity"', '"inverse"',
          '"log"', '"1/mu^2"' and '"sqrt"'. The function 'power'
          can also be used to create a power link function
          for the 'quasi' family.

variance: for all families, other than 'quasi', the variance
          function is determined by the family.  The 'quasi'
          family will accept the specifications '"constant"',
          '"mu(1-mu)"', '"mu"', '"mu^2"' and '"mu^3"' for
          the variance function.
```