GLM Tutorial in R
adapted from http://data.princeton.edu/R/glms.html

Background Info

The glm function is the basic tool for fitting generalized linear models:

> glm(formula, family, data, weights, subset, ...)

The family parameter is specific to the glm function. There are six choices of family:

<table>
<thead>
<tr>
<th>Family</th>
<th>Variance</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>gaussian</td>
<td>gaussian</td>
<td>identity</td>
</tr>
<tr>
<td>binomial</td>
<td>binomial</td>
<td>logit, probit or cloglog</td>
</tr>
<tr>
<td>poisson</td>
<td>poisson</td>
<td>log, identity or sqrt</td>
</tr>
<tr>
<td>Gamma</td>
<td>Gamma</td>
<td>inverse, identity or log</td>
</tr>
<tr>
<td>inverse.gaussian</td>
<td>inverse.gaussian</td>
<td>1/mu^2</td>
</tr>
<tr>
<td>quasi</td>
<td>user-defined</td>
<td>user-defined</td>
</tr>
</tbody>
</table>

Each of the first five choices has a variance function and one or more link functions. The last is user-defined. To use default link, all you have to do is specify the family name. To use an alternative link you must specify what you want:

> glm( formula, family=binomial(link=probit))

Get Started

1. The data needed for this tutorial are already available online. To import, copy and paste into R script:

   > cuse <- read.table("http://data.princeton.edu/wws509/datasets/cuse.dat",
                        + header=TRUE)

   The header was specified as TRUE to identify the first row of the dataset as the variable names

2. > Print(cuse) to verify data import and attach it to make it your default dataset.
3. First we can fit a simple linear regression where contraceptive use depends on the following factors: age, education, and wantsMore). This will add lrfit values to your Environment window on the right.

```r
> lrfit <- glm( cbind(using, notUsing) ~ + age + education + wantsMore , family = binomial)
```

4. This action allowed us to assign the glm function's value to an object called lrfit (logistic regression fit). The first argument of the function is a model formula, which defines the response and linear predictor.

We used binomial data, in which the response can be either a vector or a matrix with two columns.

- If the response is a vector it can be numeric with 0 for failure and 1 for success, or a factor with the first level representing "failure" and all others representing "success". In these cases R generates a vector of ones to represent the binomial denominators.
Alternatively, the response can be a matrix where the first column is the number of "successes" and the second column is the number of "failures". In this case R adds the two columns together to produce the correct binomial denominator.

Because the latter approach is the right one for us, we used the function `cbind` to create a matrix by binding the column vectors containing the numbers using and not using contraception.

Following the special symbol `~` that separates the response from the predictors, we have a standard Wilkinson-Rogers model formula. In this case we are specifying main effects of age, education and `wantsMore`. Because all three predictors are categorical variables, they are treated automatically as factors, as you can see by inspecting the results:

Call:

```r
> glm(formula = cbind(using, notUsing) ~ age + education + wantsMore, 
    family = binomial)
```

```
Coefficients:
(Intercept) age25-29 age30-39 age40-49 educationlow wantsMoreeyes
-0.8082  0.3894  0.9086  1.1892  -0.3250  -0.8330

Degrees of Freedom:  15  Total (i.e. Null):  10  Residual

Null Deviance:  165.8
Residual Deviance: 29.92  AIC: 113.4
```

5. R sorts the levels of a factor in alphabetical order. Because <25 comes before 25-29, 30-39, and 40-49, it has been picked as the reference cell for age. Similarly, high is the reference cell for education because high comes before low! Finally, R picked no as the base for `wantsMore`.

We can change (1) use `relevel` to change the base category, or (2) define your own indicator variables. For example, let's define indicators for women with high education and women who want no more children:

```r
> noMore <- wantsMore == "no"
> hiEduc <- education == "high"
```
Notice this edits your values:

![Image of R environment and data values]

Now try the model again and compare:

```r
> glm(cbind(using, notUsing) ~ age + hiEduc + noMore, family=binomial)
```

The residual deviance of 29.92 on 10 d.f. is highly significant:

```r
> 1 - pchisq(29.92, 10)
```

6. Now, test a model that introduces an interaction between age and desire for no more children

```r
> lrfit <- glm( cbind(using, notUsing) ~ age * noMore + hiEduc, family=binomial)
> lrfit
```
7. Note how R built the interaction terms automatically, and even came up with sensible labels for them. The model's deviance of 12.63 on 7 d.f. is not significant at the conventional five per cent level, so we have no evidence against this model.

To obtain more detailed information about this fit try the summary function:

```r
> summary(lrfit)
```

```
Call:  glm(formula = cbind(using, notUsing) ~ age * noMore + hiEduc, 
        family = binomial)

Deviance Residuals:
       Min          1Q       Median          3Q         Max
-1.30207     -0.86163     -0.05286     0.81945     1.73851

Coefficients:  Estimate Std. Error z value Pr(>|z|)
(Intercept)         1.80317    0.18018   10.008  < 2e-16 ***
age25-29             0.39460    0.28245    1.3959     0.00013 **
age30-39             0.54666    0.28442    1.9255     0.05497 **
age40-49             0.57952    0.34742    1.6808     0.09330
noMoreTRUE           0.06622    0.33871    0.0698     0.49430
hiEducTRUE           0.34685    0.12377    2.8083     0.00509 **
age25-29:noMoreTRUE  0.25918    0.40975    0.6338     0.527064
age30-39:noMoreTRUE  1.12066    0.37404    2.975    0.00289 **
age40-49:noMoreTRUE  1.36167    0.48433    2.8111     0.00453 **

Dispersion parameter for binomial family taken to be 1
Null deviance: 165.77 on 15 degrees of freedom
Residual deviance: 12.63 on 7 degrees of freedom
AIC: 102.1
```

---

> `lrfit <- glm(cbind(using, notUsing) ~ age * noMore + hiEduc, family=binomial)`

> `lrfit`
8. R follows the popular custom of flagging significant coefficients with one, two or three stars depending on their p-values. Try \texttt{plot(lrfit)}. You get the same plots as in a linear model, but adapted to a generalized linear model; for example the residuals plotted are deviance residuals (the square root of the contribution of an observation to the deviance, with the same sign as the raw residual).

The functions that can be used to extract results from the fit include:

- residuals or resid, for the deviance residuals
- fitted or fitted.values, for the fitted values (estimated probabilities)
- predict, for the linear predictor (estimated logits)
- coef or coefficients, for the coefficients, and
- deviance, for the deviance.

That concludes the tutorial on glm.