# Multinomial Logit Models with R

```
> rm(list=ls()); options(scipen=999) # To avoid scientific notation
> # install.packages("mlogit", dependencies=TRUE) # Only need to do this once
> library(mlogit) # Load the package every time

Loading required package: Formula
Loading required package: maxLik
Loading required package: miscTools

Please cite the 'maxLik' package as:
Henningsen, Arne and Toomet, Ott (2011). maxLik: A package for maximum likelihood
estimation in R. Computational Statistics 26(3), 443-458. DOI 10.1007/s00180-010-
0217-1.

If you have questions, suggestions, or comments regarding the 'maxLik' package,
please use a forum or 'tracker' at maxLik's R-Forge site:
https://r-forge.r-project.org/projects/maxlik/

> math =
read.table("http://www.utstat.utoronto.ca/~brunner/data/legal/mathcat.data.txt")
> head(math)

   hsgpa hsengl hscalc   course passed outcome
1  78.0     80    Yes Mainstrm     No  Failed
2  66.0     75    Yes Mainstrm    Yes  Passed
3  80.2     70    Yes Mainstrm    Yes  Passed
4  81.7     67    Yes Mainstrm    Yes  Passed
5  86.8     80    Yes Mainstrm    Yes  Passed
6  76.7     75    Yes Mainstrm    Yes  Passed
```

The explanatory vars can be characteristics of the individual case (individual-specific), or of the alternative (alternative-specific) -- that is, the value of the response variable.

The `mlogit` function requires its own special type of data frame, and there are two data formats: ``wide" and ``long." When there are individual-specific variables and lots of individuals, the wide format may be preferable, and we'll have n rows, which is what we're accustomed to. But if there are response-specific covariates, each such variable requires a separate column for each value of the response variable.

The `mlogit.data` function converts ordinary data frames to a type required by `mlogit`. I can only make the long format work.

```
> # Try a simple logistic regression.
> # First make a data frame with just what we need.
> math0 = math[,c(1,5)]; head(math0)
  hsgpa passed
1  78.0     No
2  66.0    Yes
3  80.2    Yes
4  81.7    Yes
5  86.8    Yes
6  76.7    Yes
> # Make an mlogit data frame in long format
> long0 = mlogit.data(math0,shape="wide",choice="passed")
> head(long0)
      hsgpa passed chid alt
1.No   78.0   TRUE    1  No
1.Yes  78.0  FALSE    1 Yes
2.No   66.0  FALSE    2  No
2.Yes  66.0   TRUE    2 Yes
3.No   80.2  FALSE    3  No
3.Yes  80.2   TRUE    3 Yes
```

Model description (formula) is more complex than for `glm`, because the models are more complex. Have the `mformula` function. It provides for individual-specific variables (the kind we use) and two kinds of alternative-specific variables. The user can provide 3 parts, separated by vertical bars. The first and third are alternative-specific. If we stick to individual-specific vars, we can leave off the last, like this:

```
> simple0 = mlogit(passed ~ 0 | hsgpa, data=long0)
```

```
> simple0 = mlogit(passed ~ 0 | hsgpa, data=long0); summary(simple0)


Call:
mlogit(formula = passed ~ 0 | hsgpa, data = long0, method = "nr",
    print.level = 0)

Frequencies of alternatives:
     No     Yes
0.40102 0.59898

nr method
5 iterations, 0h:0m:0s
g'(-H)^-1g = 0.000119
successive function values within tolerance limits

Coefficients :
                   Estimate Std. Error t-value            Pr(>|t|)
Yes:(intercept) -15.210112   1.998398 -7.6112 0.00000000000002709 ***
Yes:hsgpa         0.197734   0.025486  7.7587 0.00000000000000866 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Log-Likelihood: -221.72
McFadden R^2:  0.16436
Likelihood ratio test : chisq = 87.221 (p.value = < 0.00000000000000222)
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Log-Likelihood: -221.72
McFadden R^2:  0.16436
Likelihood ratio test : chisq = 87.221 (p.value = < 2.22e-16)
> # Compare ordinary logistic regression
> summary(glm(passed~hsgpa,family=binomial,data=math))

Call:
glm(formula = passed ~ hsgpa, family = binomial, data = math)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.5152  -1.0209   0.4435   0.9321   2.1302

Coefficients:
             Estimate Std. Error z value            Pr(>|z|)
(Intercept) -15.21013    1.99832  -7.611 0.00000000000002710 ***
hsgpa         0.19773    0.02548   7.759 0.00000000000000856 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 530.66  on 393  degrees of freedom
Residual deviance: 443.43  on 392  degrees of freedom
AIC: 447.43

Number of Fisher Scoring iterations: 4
```

```
> anova(glm(passed~hsgpa,family=binomial,data=math))

Analysis of Deviance Table

Model: binomial, link: logit

Response: passed

Terms added sequentially (first to last)


      Df Deviance Resid. Df Resid. Dev
NULL                    393     530.66
hsgpa  1   87.221       392     443.43

> # Compare G^2 = 87.221 from mlogit

> # Excellent. Now try simple regression with a 3-category outcome.
> # I think I have to make an mlogit data frame with just the vars I want.
> # First try to make reference category of outcome Failed.
> # Setting contrasts had no effect. Change the alphabetical order.

> outcome = as.character(math$outcome)
> for(j in 1:length(outcome))
+      {if(outcome[j]=='Disappeared') outcome[j]='Gone'}
> math$outcome = factor(outcome); table(outcome)
outcome
Failed   Gone Passed
    61     97    236

> math1 = math[,c(1,6)] # All rows, cols 1 and 6
> long1 = mlogit.data(math1,shape="wide",choice="outcome")
> head(long1)
         hsgpa outcome chid    alt
1.Failed    78    TRUE    1 Failed
1.Gone      78   FALSE    1   Gone
1.Passed    78   FALSE    1 Passed
2.Failed    66   FALSE    2 Failed
2.Gone      66   FALSE    2   Gone
2.Passed    66    TRUE    2 Passed

> head(math)
  hsgpa hsengl hscalc   course passed outcome
1  78.0     80    Yes Mainstrm     No  Failed
2  66.0     75    Yes Mainstrm    Yes  Passed
3  80.2     70    Yes Mainstrm    Yes  Passed
4  81.7     67    Yes Mainstrm    Yes  Passed
5  86.8     80    Yes Mainstrm    Yes  Passed
6  76.7     75    Yes Mainstrm    Yes  Passed
```

```
> simple1 = mlogit(outcome ~ 0 | hsgpa, data=long1)
> summary(simple1)

Call:
mlogit(formula = outcome ~ 0 | hsgpa, data = long1, method = "nr",
    print.level = 0)

Frequencies of alternatives:
 Failed    Gone  Passed
0.15482 0.24619 0.59898

nr method
5 iterations, 0h:0m:0s
g'(-H)^-1g = 1.09E-05
successive fonction values within tolerance limits

Coefficients :
                     Estimate Std. Error t-value  Pr(>|t|)
Gone:(intercept)     1.904226   2.744979  0.6937    0.4879
Passed:(intercept) -13.393056   2.570453 -5.2104 1.884e-07 ***
Gone:hsgpa          -0.018816   0.035775 -0.5260    0.5989
Passed:hsgpa         0.186437   0.033018  5.6465 1.637e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Log-Likelihood: -326.96
McFadden R^2:  0.11801
Likelihood ratio test : chisq = 87.497 (p.value = < 2.22e-16)
```

$$\log\left(\frac{\pi_1}{\pi_3}\right) = \beta_{0,1} + \beta_{1,1}x_1 + \ldots + \beta_{p-1,1}x_{p-1}$$

$$\log\left(\frac{\pi_2}{\pi_3}\right) = \beta_{0,2} + \beta_{1,2}x_1 + \ldots + \beta_{p-1,2}x_{p-1}$$

```
> # Estimate probabilities for a student with HSGPA = 90
```

$$\pi_1 = \frac{e^{L_1}}{1 + e^{L_1} + e^{L_2}}$$

$$\pi_2 = \frac{e^{L_2}}{1 + e^{L_1} + e^{L_2}}$$

$$\pi_3 = \frac{1}{1 + e^{L_1} + e^{L_2}}$$

```
> betahat1 = simple1$coefficients; betahat1
  Gone:(intercept) Passed:(intercept)         Gone:hsgpa       Passed:hsgpa
        1.90422575       -13.39305637        -0.01881621         0.18643711
attr(,"fixed")
  Gone:(intercept) Passed:(intercept)         Gone:hsgpa       Passed:hsgpa
             FALSE               FALSE              FALSE              FALSE
> gpa = 90
> L1 = betahat1[1] + betahat1[3]*gpa  # Gone
> L2 = betahat1[2] + betahat1[4]*gpa  # Passed
> denom = 1+exp(L1)+exp(L2)
> pihat1 = exp(L1)/denom   # Gone
> pihat2 = exp(L2)/denom   # Passed
> pihat3 = 1/denom         # Failed
> rbind(pihat1,pihat2,pihat3)
       Gone:(intercept)
pihat1       0.03883621
pihat2       0.92970789
pihat3       0.03145590
```

```
> # More interesting full model.
> # Make the mlogit data frame, without passed.
> head(math[,c(1:4,6)])
  hsgpa hsengl hscalc   course outcome
1  78.0     80    Yes Mainstrm  Failed
2  66.0     75    Yes Mainstrm  Passed
3  80.2     70    Yes Mainstrm  Passed
4  81.7     67    Yes Mainstrm  Passed
5  86.8     80    Yes Mainstrm  Passed
6  76.7     75    Yes Mainstrm  Passed

> long = mlogit.data(math[,c(1:4,6)],shape="wide",choice="outcome")
> # Make Mainstream the reference category for course, with nice labels.
> contrasts(long$course) = contr.treatment(3,base=3)
> colnames(contrasts(long$course)) = c("Catch-up","Elite")
> contrasts(long$course)
         Catch-up Elite
Catch-up        1     0
Elite           0     1
Mainstrm        0     0
> # Note that setting the contrasts in math does not work.
> # mlogit.data converts them back to the default!

> fullmod = mlogit(outcome ~ 0 | hsgpa+hsengl+hscalc+course, data=long)
> summary(fullmod)

Call:
mlogit(formula = outcome ~ 0 | hsgpa + hsengl + hscalc + course,
    data = long, method = "nr", print.level = 0)

Frequencies of alternatives:
 Failed    Gone  Passed
0.15482 0.24619 0.59898

nr method
5 iterations, 0h:0m:0s
g'(-H)^-1g = 0.000216
successive function values within tolerance limits

Coefficients :
                         Estimate  Std. Error t-value      Pr(>|t|)
Gone:(intercept)        1.8899724   2.8959975  0.6526       0.51400
Passed:(intercept)    -13.6325290   2.7562888 -4.9460 0.00000075765 ***
Gone:hsgpa             -0.0079779   0.0413277 -0.1930       0.84693
Passed:hsgpa            0.2157706   0.0382179  5.6458 0.00000001644 ***
Gone:hsengl            -0.0067241   0.0251049 -0.2678       0.78882
Passed:hsengl          -0.0399811   0.0228733 -1.7479       0.08047 .
Gone:hscalcYes         -0.3902775   0.6742796 -0.5788       0.56272
Passed:hscalcYes        1.0009683   0.8215247  1.2184       0.22306
Gone:courseCatch-up     0.6834686   0.5560854  1.2291       0.21905
Passed:courseCatch-up  -0.4086564   0.6339142 -0.6447       0.51915
Gone:courseElite       -1.3831859   0.8700722 -1.5897       0.11189
Passed:courseElite      0.1946275   0.5664312  0.3436       0.73114
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Log-Likelihood: -312.26
McFadden R^2:  0.15766
Likelihood ratio test : chisq = 116.89 (p.value = < 0.000000000000000222)
```

```
> # Test Course controlling for HS variables
> nocourse = mlogit(outcome ~ 0 | hsgpa+hsengl+hscalc, data=long)
> anova(nocourse,fullmod)
Error in UseMethod("anova") :
  no applicable method for 'anova' applied to an object of class "mlogit"
> # Oh well.
> G2 = -2 * as.numeric(nocourse$logLik - fullmod$logLik); G2
[1] 11.86122
> pval = 1-pchisq(G2,df=4) # Two betas for each dummy variable.
> pval
[1] 0.01841369

> # Let's keep course and hsgpa.  Do we need hsengl and hscalc?
> coursegpa = mlogit(outcome ~ 0 | hsgpa+course, data=long)
> G2 = -2 * as.numeric(coursegpa$logLik - fullmod$logLik); G2
[1] 8.457276
> pval = 1-pchisq(G2,df=4) # df=4 again
> pval
[1] 0.07619288
```

Conclusion: Let's keep just `course` and `hsgpa`.

```
> summary(coursegpa)


Call:
mlogit(formula = outcome ~ 0 | hsgpa + course, data = long, method = "nr",
    print.level = 0)

Frequencies of alternatives:
 Failed    Gone  Passed
0.15482 0.24619 0.59898

nr method
5 iterations, 0h:0m:0s
g'(-H)^-1g = 0.00016
successive function values within tolerance limits

Coefficients :
                      Estimate Std. Error t-value      Pr(>|t|)
Gone:(intercept)      1.363464   2.818403  0.4838       0.62855
Passed:(intercept)  -13.264833   2.611242 -5.0799 0.00000037765 ***
Gone:hsgpa           -0.012635   0.036570 -0.3455       0.72971
Passed:hsgpa          0.184952   0.033426  5.5332 0.00000003144 ***
Gone:courseCatch-up   0.838789   0.501793  1.6716       0.09461 .
Passed:courseCatch-up -0.707533   0.590061 -1.1991       0.23049
Gone:courseElite      -1.322791   0.857624 -1.5424       0.12298
Passed:courseElite     0.357558   0.549697  0.6505       0.51539
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Log-Likelihood: -316.49
McFadden R^2:  0.14625
Likelihood ratio test : chisq = 108.43 (p.value = < 0.000000000000000222)
```