**Multinomial regression models**

**Multinomial regression basics**

Introduction to the multinomial distribution

The binomial distribution is used when there are two categories for a response. For example, a success or failure may be the response observed over N possible trials each with the same probability of success. The multinomial distribution is the extension of the binomial distribution to J possible categories for a response.

Suppose Y = j denotes the response category for j = 1, …, J. The multinomial probability distribution is



where n is the total number of trials, nj is the number of trials with Y = j, and πj = P(Y = j). Note that .

If there are N observations of the n trials each with the same πj for j = 1, …, J, the likelihood function becomes:



where nr is the total number of trials for the rth observation, nrj is the number of trials with Y = j for the rth observation, and πj = P(Y = j).

Introduction to multinomial regression

Multinomial regression models allow for πj to be a function of independent (explanatory) variables in much the same way that logistic regression models allow for it. These models are used to estimate the probability that a response will fall into one of J possible categories. For our purposes, the categories are the populations for which we want to classify our observations into.

Suppose there are J categories for the response variable with corresponding probabilities π1, π2, …, πJ. Using the first category as a “baseline”, we can form “baseline category logits” as log(πj/π1) for j = 2, …, J.

When J = 2, we have log(π2/π1) = log(π2/(1-π2)), which is equivalent to log(π/(1-π)) in logistic regression with π = π2.

When there is only one independnet variable x, we can form the multinomial regression model of

log(πj/π1) = βj0 + βj1x for j = 2, …, J

One can easily compare other categories so that category 1 is not always used. For example, suppose you would like to compare category 2 to 3 for J ≥ 3. Then

log(π2/π1) – log(π3/π1) = log(π2) – log(π3) = log(π2/π3)

and

β20 + β21x – β30 – β31x = β20 – β30 + x(β21 – β31)

For more than one independent variable, the model becomes:

log(πj/π1) = βj0 + βj1x1 + … + βjpxp for j = 2, …, J

What is πj only? Consider the case of one explanatory variable x again:

We can re-write the model as . Noting that , we have



Thus,



Also, we can now find that



for j = 2, …, J.

Parameters are estimated using maximum likelihood estimation. For a sample of size N, the likelihood function is essentially the same as shown earlier, but with πj as given above based on the model. Iterative numerical procedures are used then to find the parameter estimates. The multinom() function from the nnet package (within the default installation of R) performs the necessary computations.

Example: Wheat kernels (WheatMultReg.R, wheat.csv)

Wheat producers want to identify kernels that are in poor condition after being harvested. To facilitate this identification process, categorization systems have been developed to partition kernels into different categories. For this example, we will look at the categories of “healthy”, “sprout”, or “scab”. In summary,

* Healthy is the preferred condition because these kernels have not been damaged
* Sprout is less preferred than healthy because they have reduced weight and poorer flour quality
* Scab is less preferred than healthy because they come from plants that have been infected by a disease and have undesirable qualities in their appearance

Example of scab kernels from “Wheat Disease Identification”:

A close-up of wheat

Description automatically generated

Ideally, it would be preferred to make these categorizations for each kernel through using an automated process. To test a new system out, 275 wheat kernels were classified by human examination (assumed to be perfect). The automated system uses information about the class of the wheat kernel (soft red winter or hard red winter) and measurements for density, hardness, size, weight, and moisture for the kernel. Below is part of the data and plots of the data.

> wheat <- read.csv(file = "C:\\data\\wheat.csv", stringsAsFactors = TRUE)

> head(wheat, n = 3)

class density hardness size weight moisture type

1 hrw 1.349253 60.32952 2.30274 24.6480 12.01538 Healthy

2 hrw 1.287440 56.08972 2.72573 33.2985 12.17396 Healthy

3 hrw 1.233985 43.98743 2.51246 31.7580 11.87949 Healthy

> tail(wheat, n = 3)

class density hardness size weight moisture type

273 srw 0.8491887 34.06615 1.40665 12.0870 11.92744 Scab

274 srw 1.1770230 60.97838 1.05690 9.4800 12.24046 Scab

275 srw 1.0305543 -9.57063 2.05691 23.8185 12.64962 Scab

> nrow(wheat2)

[1] 275

The stringsAsFactors = TRUE argument value is very important in read.csv(). R automatically will recognize any variable with letter values as a factor class type. This is needed here because the class and type variables are categorical.

> options(width = 60)

> summary(wheat2)

class density hardness size

hrw:143 Min. :0.740 Min. :-44.08 Min. :0.600

srw:132 1st Qu.:1.135 1st Qu.: 0.69 1st Qu.:1.890

Median :1.210 Median : 24.47 Median :2.230

Mean :1.189 Mean : 25.56 Mean :2.205

3rd Qu.:1.270 3rd Qu.: 45.60 3rd Qu.:2.510

Max. :1.650 Max. :111.93 Max. :4.310

weight moisture type

Min. : 8.53 Min. : 6.49 Healthy:96

1st Qu.:21.98 1st Qu.: 9.54 Scab :83

Median :27.61 Median :11.91 Sprout :96

Mean :27.50 Mean :11.19

3rd Qu.:32.88 3rd Qu.:12.54

Max. :46.33 Max. :14.51





> wheat3 <- data.frame(kernel = 1:nrow(wheat), wheat[,2:6], class.new = ifelse(test = wheat$class == "hrw", yes = 0, no = 1))

> save <- princomp(formula = ~ density + hardness + size +

weight + moisture + class.new, data = wheat3,

cor = TRUE, scores = TRUE)

> summary(save, loadings = TRUE, cutoff = 0.0)

Importance of components:

Comp.1 Comp.2 Comp.3

Standard deviation 1.4718974 1.3126735 0.9593345

Proportion of Variance 0.3610804 0.2871853 0.1533871

Cumulative Proportion 0.3610804 0.6482656 0.8016527

Comp.4 Comp.5 Comp.6

Standard deviation 0.8454985 0.5333147 0.43679671

Proportion of Variance 0.1191446 0.0474041 0.03179856

Cumulative Proportion 0.9207973 0.9682014 1.00000000

Loadings:

Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6

density 0.286 0.306 0.622 0.650 0.044 0.114

hardness -0.361 0.237 0.660 -0.525 -0.186 -0.260

size 0.440 0.461 -0.087 -0.417 -0.235 0.598

weight 0.558 0.327 -0.157 -0.135 0.158 -0.717

moisture 0.360 -0.493 0.349 -0.331 0.604 0.174

class.new 0.391 -0.536 0.155 0.004 -0.720 -0.132

A diagram of a graph

Description automatically generated with medium confidence A diagram of different colored circles

Description automatically generated

A diagram of a graph

Description automatically generated A graph of a graph with numbers and lines

Description automatically generated with medium confidence

Comments:

* Scab kernels generally have smaller density, size, and weight values
* Healthy kernels may have higher densities
* There is much overlap for healthy and sprout kernels
* The moisture content appears to be dependent on hard or soft red winter wheat class
* Healthy kernels tend to have more higher positive PC #3 values as compared to sprout kernels which tend to have more lower negative PC #3 values
* It is doubtful that we will be able to get a 100% accuracy in our classifications due to the overlap between the populations; however, we should expect some success due to the amount of separation which does exist.

I would like to estimate the following model:

log(πj/π1) = βj0 + βj1density + … + βj6class for j = 2, 3

What does R use for j = 1, 2, and 3? Again, R always puts the levels of a categorical variable in a numerical/alphabetical ordering (0, 1, 2, …, 9, …, a, A, b, B, …, z, Z). This can be seen by using the levels() function:

> levels(wheat2$type)

[1] "Healthy" "Scab" "Sprout"

Thus, j = 1 is healthy, j = 2 is scab, and j = 3 is sprout.

Below is how to estimate a multinomial regression model using the multinom() function of the nnet package (in default installation of R):

> library(nnet)

> mod.fit <- multinom(formula = type ~ class + density +

hardness + size + weight + moisture, data = wheat2)

# weights: 24 (14 variable)

initial value 302.118379

iter 10 value 234.991271

iter 20 value 192.127549

final value 192.112352 converged

> summary(mod.fit)

Call: multinom(formula = type ~ class + density + hardness + size + weight + moisture, data = wheat2)

Coefficients:

(Intercept) classsrw density hardness size

Scab 30.54650 -0.6481277 -21.59715 -0.01590741 1.0691139

Sprout 19.16857 -0.2247384 -15.11667 -0.02102047 0.8756135

weight moisture

Scab -0.2896482 0.10956505

Sprout -0.0473169 -0.04299695

Std. Errors:

(Intercept) classsrw density hardness size

Scab 4.289865 0.6630948 3.116174 0.010274587 0.7722862

Sprout 3.767214 0.5009199 2.764306 0.008105748 0.5409317

weight moisture

Scab 0.06170252 0.1548407

Sprout 0.03697493 0.1127188

Residual Deviance: 384.2247

AIC: 412.2247

> names(mod.fit)

[1] "n" "nunits" "nconn"

[4] "conn" "nsunits" "decay"

[7] "entropy" "softmax" "censored"

[10] "value" "wts" "convergence"

[13] "fitted.values" "residuals" "lev"

[16] "call" "terms" "weights"

[19] "deviance" "rank" "lab"

[22] "coefnames" "vcoefnames" "contrasts"

[25] "xlevels" "edf" "AIC"

> head(mod.fit$fitted.values) #pi.hats

Healthy Scab Sprout

1 0.8552110 0.046396827 0.09839221

2 0.7492553 0.021572158 0.22917255

3 0.5172800 0.068979903 0.41374011

4 0.8982064 0.006740716 0.09505287

5 0.5103245 0.176260796 0.31341473

6 0.7924907 0.015304122 0.19220522

> class(mod.fit)

[1] "multinom" "nnet"

> methods(class = multinom)

[1] add1 anova coef confint

[5] drop1 extractAIC logLik model.frame

[9] predict print summary vcov

see '?methods' for accessing help and source code

Notice that class has two levels as well:

> levels(wheat2$class)

[1] "hrw" "srw"

R creates an indicator variable for it so that classsrw = 1 for soft red winter wheat and classsrw = 0 for hard red winter wheat. The reason why the 0 and 1 assignments are not reversed is because R always treats the first level of a qualitative independent variable as the “base” level. For example, if there was a four level qualitative variable (levels A, B, C, and D), there would be three indicator variables coded as

|  |  |  |  |
| --- | --- | --- | --- |
|  | Indicator variables | | |
| Levels | x1 | x2 | x3 |
| A | 0 | 0 | 0 |
| B | 1 | 0 | 0 |
| C | 0 | 1 | 0 |
| D | 0 | 0 | 1 |

The estimated multinomial regression model is



and



We can use the Anova() function to perform LRTs:

> library(car)

> Anova(mod.fit)

Analysis of Deviance Table (Type II tests)

Response: type

LR Chisq Df Pr(>Chisq)

class 0.964 2 0.6175

density 90.555 2 < 2.2e-16 \*\*\*

hardness 7.074 2 0.0291 \*

size 3.211 2 0.2008

weight 28.230 2 7.411e-07 \*\*\*

moisture 1.193 2 0.5506

---

Signif. codes:

0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

The LRTs are of the form:

H0: β2r = β3r = 0

Ha: Not all equal to 0

for variable r. Which corresponding independent variables lead to a rejection of the null hypothesis?

**Multinomial regression for prediction**

Again, our purpose of examining multinomial regression models is to use the model to predict if an observation is from one of two populations. This prediction begins by examining the estimated probabilities for each category (population). Written in terms of only one independent variable, we have:

 and  for j = 2, …, J

for each observation. Commonly, one uses the following criteria then to classify an observation:

Classify the observation into the population corresponding to the largest estimated probability. For example, if  for j = 2, …, J, then the corresponding observation is classified as coming from population #1.

For the one independent variable case, one can visualize how the classifications are done. For example, consider the model

log(π2/π1) = 29.20 – 24.42x and

log(π3/π1) = 18.84 – 15.24x

Below is a plot of the model (see MultinomialModelPlot.R for code):



The plot shows for what values of x that π1, π2, or π3 would be the largest:

* When x < 1.129, π2 > π1 and π2 > π3, so classify a corresponding observation as being from population #2.
* When x > 1.236, π1 > π2 and π1 > π3, so classify a corresponding observation as being from population #1.
* When x > 1.129 and x < 1.236, π3 > π2 and π3 > π1, so classify a corresponding observation as being from population #3.

An estimated multinomial regression model could be used in the same way.

Example: Wheat kernels (WheatMultReg.R, wheat.csv)

Using the model, we can obtain estimates of the probabilities for each kernel type. Using the formula for healthy, here’s the calculation for the first observation.



Shown below is how the  equations can be programmed into R for observation #1:

> x.vec <- c(1,0,as.numeric(wheat2[1,2:6]))

> round(x.vec, 4)

[1] 1.00 0.00 1.35 60.33 2.30 24.65 12.02

> beta.hat <- coefficients(mod.fit)

> scab.part <- exp(sum(beta.hat[1,]\*x.vec))

> sprout.part <- exp(sum(beta.hat[2,]\*x.vec))

> pi.hat.scab <- scab.part/(1+scab.part+sprout.part)

> pi.hat.sprout <- sprout.part/(1+scab.part+sprout.part)

> pi.hat.healthy <- 1/(1+scab.part+sprout.part)

> round(data.frame(pi.hat.healthy, pi.hat.scab,

pi.hat.sprout), 4)

pi.hat.healthy pi.hat.scab pi.hat.sprout

1 0.8552 0.0464 0.0984

The estimated probabilities can be found in R more easily as

> head(mod.fit$fitted.values)

Healthy Scab Sprout

1 0.8552110 0.046396827 0.09839221

2 0.7492553 0.021572158 0.22917255

3 0.5172800 0.068979903 0.41374011

4 0.8982064 0.006740716 0.09505287

5 0.5103245 0.176260796 0.31341473

6 0.7924907 0.015304122 0.19220522

> pi.hat <- predict(object = mod.fit, type = "probs")

> head(pi.hat)

Healthy Scab Sprout

1 0.8552110 0.046396827 0.09839221

2 0.7492553 0.021572158 0.22917255

3 0.5172800 0.068979903 0.41374011

4 0.8982064 0.006740716 0.09505287

5 0.5103245 0.176260796 0.31341473

6 0.7924907 0.015304122 0.19220522

> classify <- predict(object = mod.fit, type = "class")

> head(classify)

[1] Healthy Healthy Healthy Healthy Healthy Healthy

Levels: Healthy Scab Sprout

To help relate the parallel coordinates plot to these estimated probabilities, consider kernel #269 highlighted below:



The observed values and the estimated probabilities for this kernel are:

> wheat[269,]

class density hardness size weight moisture type

269 srw 0.9343233 48.66988 0.88496 8.532 11.81367 Scab

> predict(mod.fit, newdata = wheat[269,], type = "probs")

Healthy Scab Sprout

0.0001630974 0.9934978856 0.0063390170

> predict(mod.fit, newdata = wheat2[269,], type = "class")

[1] Scab

Levels: Healthy Scab Sprout

The plot shows that a characteristic of the scab kernels is their lower weights. This comes out in the model as seen by the very large estimated probability of being scab for kernel #269.

The overall accuracy of the classifications using resubstitution are

> summarize.class <- function(original, classify) {

class.table <- table(original, classify)

numb <- rowSums(class.table)

prop <- round(class.table/numb,4)

overall <- round(sum(diag(class.table)) /

sum(class.table), 4)

list(class.table = class.table, prop = prop,

overall.correct = overall)

}

> summarize.class(original = wheat2$type, classify =

classify)

$class.table

classify

original Healthy Scab Sprout

Healthy 74 6 16

Scab 9 64 10

Sprout 19 17 60

$prop

classify

original Healthy Scab Sprout

Healthy 0.7708 0.0625 0.1667

Scab 0.1084 0.7711 0.1205

Sprout 0.1979 0.1771 0.6250

$overall.correct

[1] 0.72

Overall, we see the model has some ability to differentiate between the different kernel types. The most problems occur with sprout kernels being classified as healthy or scab. The least problems occur with healthy kernels being classified as scab.

The classifications for new observations can be done using the predict() function:

> newobs <- wheat2[1,] #Suppose we have one new observation

(set equal to first for demonstration purposes)

> predict(mod.fit, newdata = newobs, type = "probs")

Healthy Scab Sprout

0.85289053 0.04696366 0.10014581

> predict(mod.fit, newdata = newobs, type = "class")

[1] Healthy

Levels: Healthy Scab Sprout

Cross-validation can be performed in the same manner as with the placekicking data with some modifications to the cv() function:

> cv2 <- function(model, data.set) {

N <- nrow(data.set)

#Determine number of levels in response variable

# Put model formula together with data set name

save.model <- model.frame(model, data = data.set)

# Response variable in data set

response <- model.response(save.model)

# Number of populations

numb.pop <- length(unique(response))

pi.hat.cv <- matrix(data = NA, nrow = N, ncol =

numb.pop)

class.cv <- character(length = N)

for(r in 1:N) {

mod.fit <- multinom(formula = model, data =

data.set[-r,], trace = FALSE)

pi.hat.cv[r,] <- predict(object = mod.fit, newdata =

data.set[r,], type = "probs")

#Need as.character() to preserve the names, otherwise

the names become numbers

class.cv[r] <- as.character(predict(object = mod.fit,

newdata = data.set[r,], type = "class"))

}

list(prob = pi.hat.cv, classify = class.cv)

}

> save.cv <- cv2(model = type ~ class + density + hardness + size + weight + moisture, data.set = wheat2)

> head(save.cv$prob)

[,1] [,2] [,3]

[1,] 0.8508365 0.047915191 0.10124831

[2,] 0.7411553 0.022128702 0.23671603

[3,] 0.5048828 0.070458065 0.42465912

[4,] 0.8958373 0.006872012 0.09729066

[5,] 0.4946104 0.183275696 0.32211391

[6,] 0.7866402 0.015647414 0.19771237

> head(save.cv$classify)

[1] "Healthy" "Healthy" "Healthy" "Healthy" "Healthy"

[6] "Healthy"

> summarize.class(original = wheat2$type, classify =

save.cv$classify)

$class.table

classify

original Healthy Scab Sprout

Healthy 71 6 19

Scab 10 60 13

Sprout 19 19 58

$prop

classify

original Healthy Scab Sprout

Healthy 0.7396 0.0625 0.1979

Scab 0.1205 0.7229 0.1566

Sprout 0.1979 0.1979 0.6042

$overall.correct

[1] 0.6873

Comments:

* The model.frame() and model.respose() functions are used to help me isolate the response variable so that I can extract the levels associated with it. The number of levels corresponds to the number of probabilities that I will obtain for each observation.
* The classifications are performed within the function rather than outside of the function as was done in the placekicking data example. If done outside of the function, I would have needed J – 1 different nested ifelse() functions to make the classification. By doing it within the function, I can do it more easily.
* The overall accuracy is a little lower than what we obtained through resubstitution. This is to be expected for the same reasons as first discussed in the DA section.

Below is a plot comparing the accuracy of a number of classification methods (code not in program).

A graph with red dots and white text

Description automatically generated

Which method is best?

**Additional considerations for multinomial regression**

* ROC curves can be constructed as well, but the definitions of sensitivity and specificity need to be extended to accommodate J > 2 populations. This area of research is not as well developed as for the J = 2 case.
* Variable selection can be performed by standard methods as when working with regression models. For example, the “best” model can be thought of as the one with the smallest Akaike’s information criteria (AIC). However, this does not address the classification accuracy of the model.
* There are many other types of regression models that can be used with multinomial responses. One popular model is a proportional odds model. This model is used when the J categories are ordered as

category 1 < category 2 < < category J

If Y denotes the category response and P(Y = j) = πj, the cumulative probability for Y is

P(Y j) = π1 + … + πj

for j = 1, …, J. Note that P(Y J) = 1. The logit of this cumulative probability is models as a function of the independent variables:



for j = 1, …, J – 1. For each j, the model compares the log odds of being in categories 1 through j vs. categories j + 1 through J. In terms of this model, the πj values can be found as π1 = , πJ = , and



for j = 2, …, J – 1. When ordering of the category response actually occurs, this model can be much better than the multinomial regression which does not take into account any ordering.

With respect to the wheat data set, there is some justification for an ordering of scab < sprout < healthy. The WheatMultReg.R program estimates the corresponding proportional odds model, and resubstitution results in the following accuracy measures:

> summarize.class(original = wheat2$type.order, classify =

classify.ord)

$class.table

classify

original Scab Sprout Healthy

Scab 58 19 6

Sprout 17 46 33

Healthy 2 26 68

$prop

classify

original Scab Sprout Healthy

Scab 0.6988 0.2289 0.0723

Sprout 0.1771 0.4792 0.3438

Healthy 0.0208 0.2708 0.7083

$overall.correct

[1] 0.6255

The overall accuracy here is somewhat smaller than what we had with the multinomial regression model.

* The multinom() function can stop its iterative numerical procedure a little earlier than it should at times when judging convergence. Increasing the number of iterations (maxit argument) and decreasing the convergence criterion (reltol argument, like epsilon used for glm()) helps to achieve convergence. Please see my Categorical Data Analysis course notes for more information.