Introduction to Statistical Learning

Notes by Tim Brown and Owen Jones

Module 9: Multinomial and Ordinal Data

Contents

1 Multinomial logit model

1.1 Big Picture- F5.1

Where we have been.

In modules 7 and 8, the theory and practice of generalised linear models were developed. This enabled estimation, confidence intervals and hypothesis testing in a wide variety of situations where data is not normally distributed. Questions of model selection, reliability, repeatability and robustness still required the distribution theory for confidence intervals and hypothesis tests. Maximum likelihood and link functions were the key. Model fit used deviance as well as Akaike Information Criterion.

Where are we going?

In module 9, the data are counts but the individual responses are vectors rather than numbers. Each response can be thought of as counting a number of random throws of balls into a fixed number of boxes, with the Binomial model being the special case with two boxes. Maximum likelihood, link functions and deviance again feature prominently.

1.2 Definition- F5.1

Multinomial logit model

Suppose $\mathbf{Y}_1, \ldots, \mathbf{Y}_n$ are independent random vectors of counts with:

$$\begin{aligned} \mathbf{Y}_i &\sim \text{multinomial}(m_i, \mathbf{p}_i) \quad \text{with } \mathbf{p}_i = (p_{i1}, \ldots, p_{iJ}) \\ \mathbb{P}(\mathbf{Y}_i = \mathbf{y}_i) &= \frac{m_i!}{y_{i1}! \cdots y_{iJ}!} p_{i1}^{y_{i1}} \cdots p_{iJ}^{y_{iJ}} \quad \text{for } \mathbf{y}_i \geq 0, \textstyle\sum_j y_{ij} = m_i \end{aligned}$$

A *multinomial logit model* supposes that

$$p_{ij} = \frac{e^{\eta_{ij}}}{\sum_{k=1}^{J} e^{\eta_{ik}}}$$

where $\eta_{ij} = \mathbf{x}_i^T \boldsymbol{\beta}_j$ for predictor variables $\mathbf{x}_1, \ldots, \mathbf{x}_n$ and parameter vectors $\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_J$.

Thus the linear predictor η_{ij} is the log of the odds of the probability p_{ij}.

The interpretation of $e^{\eta_{ij}}$ can be the rate at which an outcome of type j occurs.

For fixed i each η_{ij} can have a constant added without changing the p_{ij}, so the model is not uniquely specified. This is fixed by setting $\boldsymbol{\beta}_1 = \mathbf{0}$. This is equivalent to dividing top and bottom by $e^{\eta_{i1}}$, or alternatively replacing $\boldsymbol{\beta}_j$ by $\boldsymbol{\beta}_j - \boldsymbol{\beta}_1$.

If $J = 2$ then

$$p_{i1} = \frac{1}{1 + e^{\mathbf{x}_i^T \boldsymbol{\beta}_2}}, \quad p_{i2} = \frac{e^{\mathbf{x}_2^T \boldsymbol{\beta}_2}}{1 + e^{\mathbf{x}_2^T \boldsymbol{\beta}_2}}$$

which is just a binomial regression model with responses y_{i2}, parameters $\boldsymbol{\beta}_2$, and a logit link.

1.3 Fit by MLE- F5.1

Maximum Likelihood reigns

Multinomial logit model are fitted using maximum likelihood estimation. The resulting estimator is then consistent and asymptotically efficient. Nested models can be evaluated by comparing the difference of their deviances to a χ^2 distribution, whose degrees of freedom are the difference in the number of parameters.

1.4 Example 1996 American National Election Study - F5.1

Data description and aim

10 variable subset of the 1996 American National Election Study, part of a series by Rosenstone, Kinder, and Miller from the University of Michigan Institute for Social Research. Missing values and "don't know" responses have been listwise deleted leaving 944 observations. Respondents expressing a voting preference other than for the presidential candidates Clinton or Dole

have been removed. Of interest is the response of party affiliation, categorised as Republican, Democrat or Independent. This will be modelled using age, education and income levels.

Levels of party affiliation and income

```
library(faraway)
data(nes96)
levels(nes96$PID)

## [1] "strDem"  "weakDem" "indDem"  "indind"  "indRep"  "weakRep" "strRep"

levels(nes96$income)

##  [1] "$3Kminus"   "$3K-$5K"    "$5K-$7K"    "$7K-$9K"    "$9K-$10K"
##  [6] "$10K-$11K"  "$11K-$12K"  "$12K-$13K"  "$13K-$14K"  "$14K-$15K"
## [11] "$15K-$17K"  "$17K-$20K"  "$20K-$22K"  "$22K-$25K"  "$25K-$30K"
## [16] "$30K-$35K"  "$35K-$40K"  "$40K-$45K"  "$45K-$50K"  "$50K-$60K"
## [21] "$60K-$75K"  "$75K-$90K"  "$90K-$105K" "$105Kplus"

sPID <- nes96$PID
# recode party affiliation as Republican, Democrat or Independent
levels(sPID) <- c("Democrat","Democrat","Independent","Independent",
"Independent","Republican","Republican")
# recode income as midpoint of group to make it numerical
inca <- c(1.5,4,6,8,9.5,10.5,11.5,12.5,13.5,14.5,16,18.5,21,23.5,
27.5,32.5,37.5,42.5,47.5,55,67.5,82.5,97.5,115)
nincome <- inca[unclass(nes96$income)]
```

Plotting voting preference against education

```
table(nes96$educ, sPID)

##         sPID
##          Democrat Independent Republican
##   MS            9           3          1
##   HSdrop       29          14          9
##   HS          108          63         77
##   Coll         74          40         73
##   CCdeg        34          24         32
##   BAdeg        81          55         91
##   MAdeg        45          40         42

prop.table(table(nes96$educ, sPID), 1)

##         sPID
##            Democrat Independent Republican
##   MS     0.69230769  0.23076923 0.07692308
##   HSdrop 0.55769231  0.26923077 0.17307692
##   HS     0.43548387  0.25403226 0.31048387
##   Coll   0.39572193  0.21390374 0.39037433
##   CCdeg  0.37777778  0.26666667 0.35555556
##   BAdeg  0.35682819  0.24229075 0.40088106
##   MAdeg  0.35433071  0.31496063 0.33070866
```

```
# matplot plots one column of a matrix against another
matplot(prop.table(table(nes96$educ, sPID), 1), type="o")
```

Plotting voting preference against income and age

```
# plotting voting preference against income
matplot(prop.table(table(nincome, sPID), 1), type="o")
# plotting voting preference agains age;
# need to group age values
matplot(prop.table(table(cut(nes96$age, 6), sPID), 1),
type="o")
```

Figure 1: Voting preference versus education

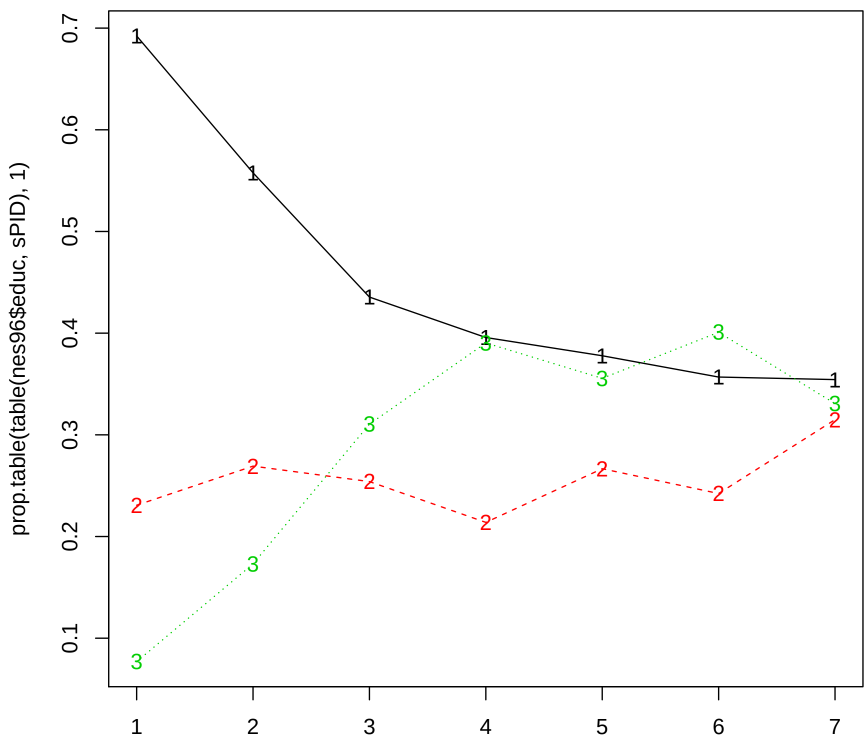
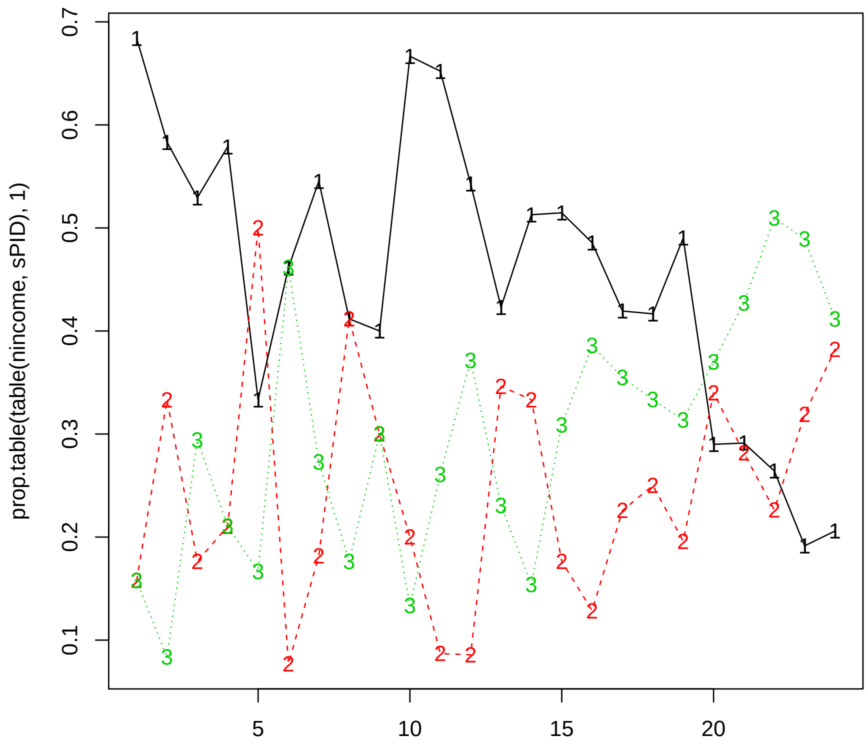


Figure 2: Voting preference versus income

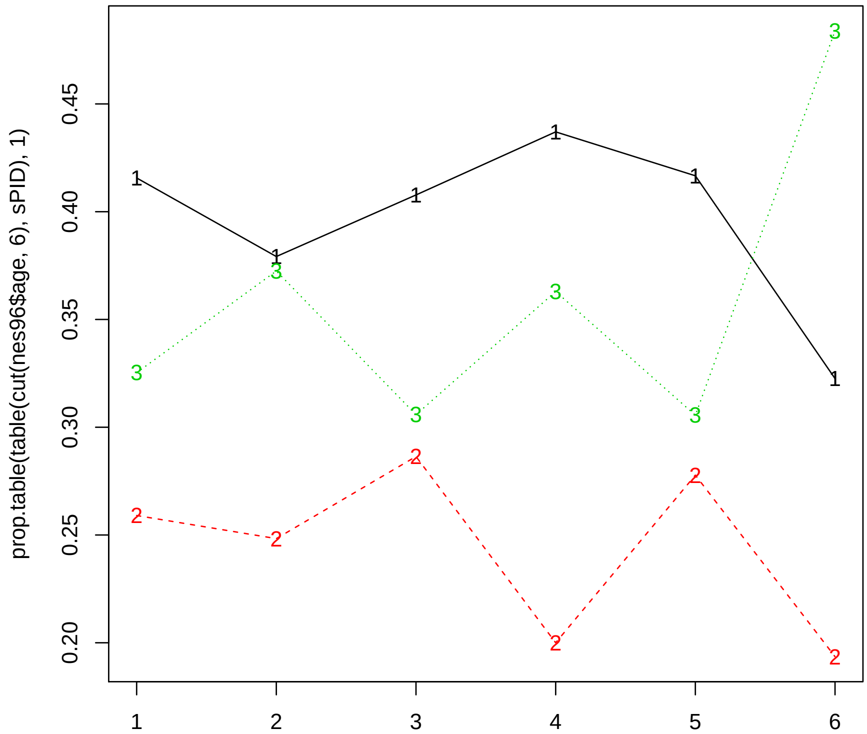



```
levels(cut(nes96$age, 6))

## [1] "(18.9,31]" "(31,43]"   "(43,55]"   "(55,67]"   "(67,79]"   "(79,91.1]"
```

Fitting a model

```
library(nnet)
mmod <- multinom(sPID ~ age + educ + nincome, nes96)

## # weights:  30 (18 variable)
## initial  value 1037.090001
## iter  10 value 990.568608
## iter  20 value 984.319052
```

Figure 3: Voting preference versus age



```
## final  value 984.166272
## converged
```

Results

```r
summary(mmod)
```

```
## Call:
## multinom(formula = sPID ~ age + educ + nincome, data = nes96)
##
## Coefficients:
##             (Intercept)          age     educ.L     educ.Q    educ.C
## Independent   -1.197260 0.0001534525 0.06351451 -0.1217038 0.1119542
## Republican    -1.642656 0.0081943691 1.19413345 -1.2292869 0.1544575
##                  educ^4     educ^5      educ^6    nincome
## Independent -0.07657336  0.1360851  0.15427826 0.01623911
## Republican  -0.02827297 -0.1221176 -0.03741389 0.01724679
##
## Std. Errors:
##             (Intercept)         age    educ.L    educ.Q    educ.C
## Independent   0.3265951 0.005374592 0.4571884 0.4142859 0.3498491
## Republican    0.3312877 0.004902668 0.6502670 0.6041924 0.4866432
##                educ^4    educ^5    educ^6     nincome
## Independent 0.2883031 0.2494706 0.2171578 0.003108585
## Republican  0.3605620 0.2696036 0.2031859 0.002881745
##
## Residual Deviance: 1968.333
## AIC: 2004.333
```

Model selection using AIC

```r
# model selection using AIC
mmodi <- step(mmod)
```

```
## Start:  AIC=2004.33
## sPID ~ age + educ + nincome
##
## trying - age
## # weights:  27 (16 variable)
## initial  value 1037.090001
## iter  10 value 988.896864
## iter  20 value 985.822223
## final  value 985.812737
## converged
## trying - educ
## # weights:  12 (6 variable)
## initial  value 1037.090001
## iter  10 value 992.269502
## final  value 992.269484
## converged
## trying - nincome
## # weights:  27 (16 variable)
## initial  value 1037.090001
## iter  10 value 1009.025560
## iter  20 value 1006.961593
## final  value 1006.955275
## converged
##           Df      AIC
## - educ     6 1996.539
## - age     16 2003.625
## <none>    18 2004.333
## - nincome 16 2045.911
## # weights:  12 (6 variable)
## initial  value 1037.090001
## iter  10 value 992.269502
## final  value 992.269484
## converged
##
## Step:  AIC=1996.54
## sPID ~ age + nincome
##
## trying - age
## # weights:  9 (4 variable)
## initial  value 1037.090001
## final  value 992.712152
## converged
## trying - nincome
## # weights:  9 (4 variable)
## initial  value 1037.090001
## final  value 1020.425203
## converged
##           Df      AIC
## - age      4 1993.424
## <none>     6 1996.539
## - nincome  4 2048.850
## # weights:  9 (4 variable)
```

```
## initial  value 1037.090001
## final  value 992.712152
## converged
##
## Step:  AIC=1993.42
## sPID ~ nincome
##
## trying - nincome
## # weights:  6 (2 variable)
## initial  value 1037.090001
## final  value 1020.636052
## converged
##           Df      AIC
## <none>     4 1993.424
## - nincome  2 2045.272
```

Results

```r
summary(mmodi)
```

```
## Call:
## multinom(formula = sPID ~ nincome, data = nes96)
##
## Coefficients:
##             (Intercept)    nincome
## Independent  -1.1749331 0.01608683
## Republican   -0.9503591 0.01766457
##
## Std. Errors:
##             (Intercept)     nincome
## Independent   0.1536103 0.002849738
## Republican    0.1416859 0.002652532
##
## Residual Deviance: 1985.424
## AIC: 1993.424
```

Model selection using Deviance

```r
# model selection using likelihood ratios
mmode <- multinom(sPID ~ age + nincome, nes96)
```

```
## # weights:  12 (6 variable)
## initial  value 1037.090001
## iter  10 value 992.269502
## final  value 992.269484
## converged
```

```r
deviance(mmode) - deviance(mmod)
```

```
## [1] 16.20642
```

```r
mmod$edf
```

```
## [1] 18
```

```r
mmode$edf
```

```
## [1] 6
```

```r
pchisq(16.206, mmod$edf - mmode$edf, lower=FALSE)
```

```
## [1] 0.181982
```

Fitted probabilities

```r
predict(mmodi, data.frame(nincome=inca), type="probs")
```

```
##     Democrat Independent Republican
## 1  0.5836466   0.1846557  0.2316977
## 2  0.5733047   0.1888271  0.2378682
## 3  0.5649837   0.1921708  0.2428455
## 4  0.5566253   0.1955183  0.2478565
## 5  0.5503347   0.1980300  0.2516353
## 6  0.5461317   0.1997045  0.2541638
## 7  0.5419219   0.2013787  0.2566993
## 8  0.5377060   0.2030524  0.2592415
## 9  0.5334846   0.2047254  0.2617901
```

```
## 10 0.5292582   0.2063972  0.2643446
## 11 0.5229106   0.2089023  0.2681871
## 12 0.5123151   0.2130684  0.2746165
## 13 0.5017076   0.2172194  0.2810730
## 14 0.4910976   0.2213511  0.2875513
## 15 0.4741402   0.2279116  0.2979482
## 16 0.4530281   0.2360027  0.3109692
## 17 0.4320800   0.2439428  0.3239772
## 18 0.4113683   0.2517021  0.3369297
## 19 0.3909623   0.2592525  0.3497852
## 20 0.3610676   0.2701312  0.3688012
## 21 0.3136199   0.2868931  0.3994870
## 22 0.2614599   0.3044513  0.4340888
## 23 0.2152314   0.3190178  0.4657508
## 24 0.1691487   0.3322310  0.4986204
```

Plot of fitted probabilities

```
matplot(predict(mmodi, data.frame(nincome=inca),
type="probs"),
type="l", ylim=c(0, .7))
matplot(prop.table(table(nincome, sPID), 1), add=TRUE)
```

2 Ordinal data

2.1 Definition and Interpretation - F5.3

Ordinal data

Suppose that we have independent observations $Y_i \in \{1, \ldots, J\}$, representing ordered categories. Put

$$p_{ij} = \mathbb{P}(Y_i = j) \text{ and } \gamma_{ij} = \mathbb{P}(Y_i \leq j).$$

We could model these data using a multinomial logistic regression with $m_i = 1$, but this ignores the ordering.

Instead we suppose that for some link function g

$$g(\gamma_{ij}) = \theta_j - \mathbf{x}_i^T \boldsymbol{\beta}$$

Note:

- g is usually the logit, probit or complementary log log link;
- $\boldsymbol{\beta}$ does not depend on j;
- the $\mathbf{x}_i^T \boldsymbol{\beta}$ term must not include an intercept.

Interpretation of ordinal model

Suppose that response Y_i is a discretised version of some continuous r.v. $\tilde{Y}_i$, where the distribution of $\tilde{Y}_i$ is given by

$$\tilde{Y}_i \sim Z + \mathbf{x}_i^T \boldsymbol{\beta} \text{ for some } Z.$$

Figure 4: Voting preference versus income with fitted values

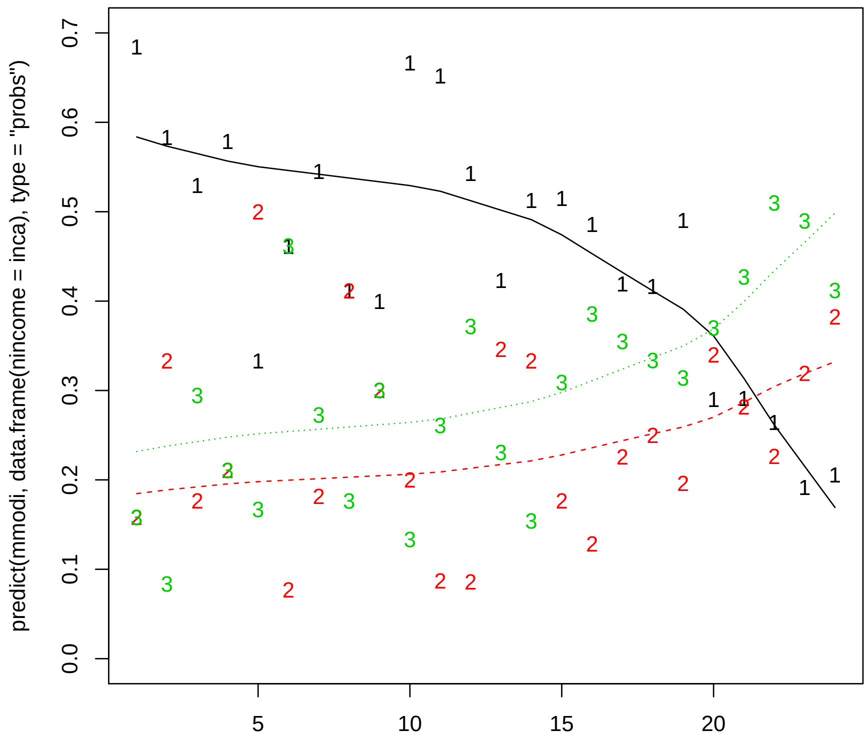
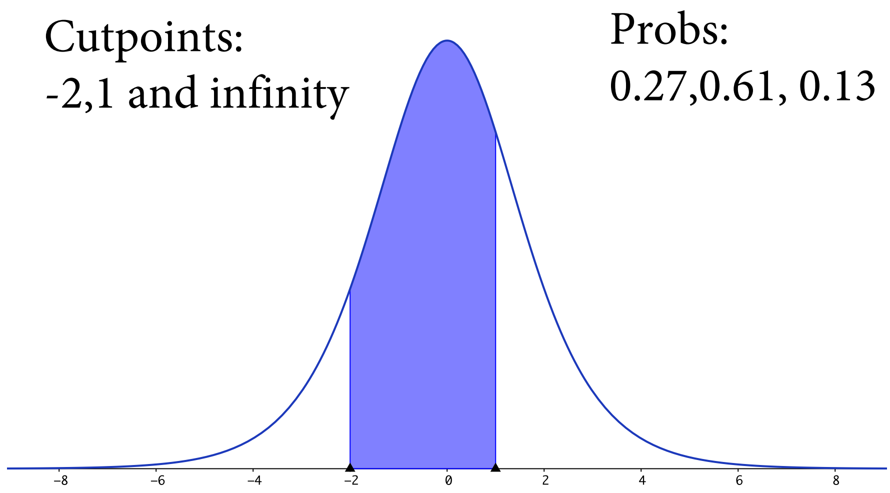


Figure 5: Standard Logistic



The θ_j are then defined by $\mathbb{P}(Y_i \leq j) = \mathbb{P}(\tilde{Y}_i \leq \theta_j)$, which gives

$$\begin{aligned}\gamma_{ij} &= \mathbb{P}(Y_i \leq j)\\ &= \mathbb{P}(\tilde{Y}_i \leq \theta_j)\\ &= \mathbb{P}(Z + \mathbf{x}_i^T \boldsymbol{\beta} \leq \theta_j)\\ &= \mathbb{P}(Z \leq \theta_j - \mathbf{x}_i^T \boldsymbol{\beta})\\ &= F_Z(\theta_j - \mathbf{x}_i^T \boldsymbol{\beta})\end{aligned}$$

Put $g = F_Z^{-1}$ to get our model. Like the multinomial logit model, we can fit an ordinal model using maximum likelihood.

2.2 Logisitic Distribution - F5.3

Logistic Distribution

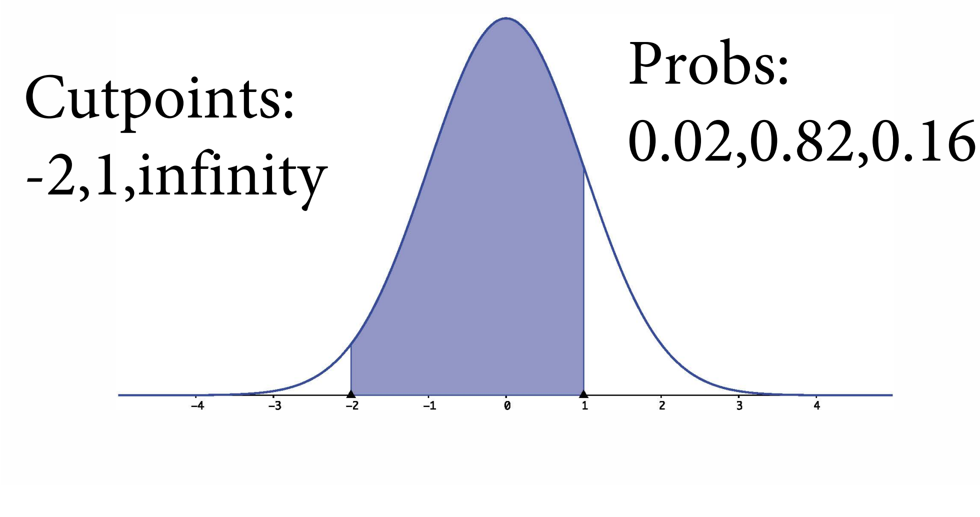
The *Standard Logistic* distribution has cdf, $F(x) = e^x/(1 + e^x)$. This is a distribution function because it's derivative is a pdf $f(x) = e^x/(1 + e^x)^2$. The mean is 0 by symmetry of the pdf. The variance is 1.8, so the probabilities are not directly comparable to standard normal ones. It is possible to introduce both scale and location parameters so that logistic distributions are possible with any mean or variance.

2.3 Latent Variable Idea - F5.3

Latent Variables

The random variable, Z, is often called a *latent variable*. Latent variables often represent some quantity that is not directly observed. An example might

Figure 6: Normal mean 0, variance 1



be *mathematical aptitude* where different people have different capacities, by virtute of experience and genetics, to do mathematics. Various tests can be given for such an aptitude but the results will never give identical conclusions. For a binary response, mean and probabilities are the same so the logistic and normal models correspond to logistic and probit links in the binomial GLM. Certain kinds of latent variable models based on the logistic distribution are commonly used in education and called *Rasch* models.

2.4 Example 1996 American National Election Study - F5.1

NES96 as Ordinal

The American National Election could be thought of as having an ordinal response if Independents are regarded as intermediate between Democrats and Republicans. An ordinal model based on the logistic or normal distribution is then possible. The observed dependence of the voter preference on the other variables can then be examined with the ordinal model. The aim is to compare the results to the more general categorical model. The R library MASS contains the commamd `polr` which does ordinal logistic regression by default.

Fitting the ordinal model

```
library(MASS)
omod <- polr(sPID ~ age + educ + nincome, nes96)
summary(omod)

## ## Re-fitting to get Hessian
```

```
## Call:
## polr(formula = sPID ~ age + educ + nincome, data = nes96)
##
## Coefficients:
##             Value Std. Error  t value
## age      0.005775   0.003887  1.48581
## educ.L   0.724087   0.384388  1.88374
## educ.Q  -0.781361   0.351173 -2.22500
## educ.C   0.040168   0.291762  0.13767
## educ^4  -0.019925   0.232429 -0.08573
## educ^5  -0.079413   0.191533 -0.41462
## educ^6  -0.061104   0.157747 -0.38735
## nincome  0.012739   0.002140  5.95187
##
## Intercepts:
##                        Value   Std. Error t value
## Democrat|Independent    0.6449  0.2435     2.6479
## Independent|Republican  1.7374  0.2493     6.9694
##
## Residual Deviance: 1984.211
## AIC: 2004.211
```

Compare to categorical

```
summary(mmod)
```

```
## Call:
## multinom(formula = sPID ~ age + educ + nincome, data = nes96)
##
## Coefficients:
##             (Intercept)          age     educ.L     educ.Q    educ.C
## Independent   -1.197260 0.0001534525 0.06351451 -0.1217038 0.1119542
## Republican    -1.642656 0.0081943691 1.19413345 -1.2292869 0.1544575
##                  educ^4     educ^5      educ^6    nincome
## Independent -0.07657336  0.1360851  0.15427826 0.01623911
## Republican  -0.02827297 -0.1221176 -0.03741389 0.01724679
##
## Std. Errors:
##             (Intercept)         age    educ.L    educ.Q    educ.C
## Independent   0.3265951 0.005374592 0.4571884 0.4142859 0.3498491
## Republican    0.3312877 0.004902668 0.6502670 0.6041924 0.4866432
##                educ^4    educ^5    educ^6     nincome
## Independent 0.2883031 0.2494706 0.2171578 0.003108585
## Republican  0.3605620 0.2696036 0.2031859 0.002881745
##
## Residual Deviance: 1968.333
## AIC: 2004.333
```

Residual deviance higher for the ordinal model - df are 10 and 18 for the two models. What happens with stepwise selection using AIC?

Model selection using AIC

```
omod2 <- step(omod)
```

```
## Start:  AIC=2004.21
## sPID ~ age + educ + nincome
##
##           Df    AIC
## - educ     6 2002.8
## <none>       2004.2
## - age      1 2004.4
## - nincome  1 2038.6
##
## Step:  AIC=2002.83
## sPID ~ age + nincome
##
##           Df    AIC
## - age      1 2001.4
## <none>       2002.8
## - nincome  1 2047.2
##
## Step:  AIC=2001.36
## sPID ~ nincome
##
##           Df    AIC
## <none>       2001.4
## - nincome  1 2045.3
```

So just as with the unordered model, voter preference is modelled on income.

Summary of selected model

```
summary(omod2)

## ## Re-fitting to get Hessian

## Call:
## polr(formula = sPID ~ nincome, data = nes96)
##
## Coefficients:
##           Value Std. Error t value
## nincome 0.01312   0.001971   6.657
##
## Intercepts:
##                        Value   Std. Error t value
## Democrat|Independent    0.2091  0.1123     1.8627
## Independent|Republican  1.2916  0.1201    10.7526
##
## Residual Deviance: 1995.363
## AIC: 2001.363
```

So for a person with income \$50,000, the fitted log odds of being Democrat is $0.2091 - 0.0132 \times 50$, and the fitted log odds of being Democrat or Independent is $1.2916 - 0.0132 \times 50$.

Model confirmation using LR Test

```
deviance(omod2) - deviance(omod)

## [1] 11.15136

omod$edf

## [1] 10

omod2$edf

## [1] 3

pchisq(11.151, omod$edf - omod2$edf, lower=FALSE)

## [1] 0.1321668
```

So we cannot reject the null hypothesis that all of the parameters in the full model, other than those in the selected model, are 0. Hence the likelihood ratio test confirms the adequacy of the selected model. Parsimony suggests this is the best model.

Comparison of fitted probabilities

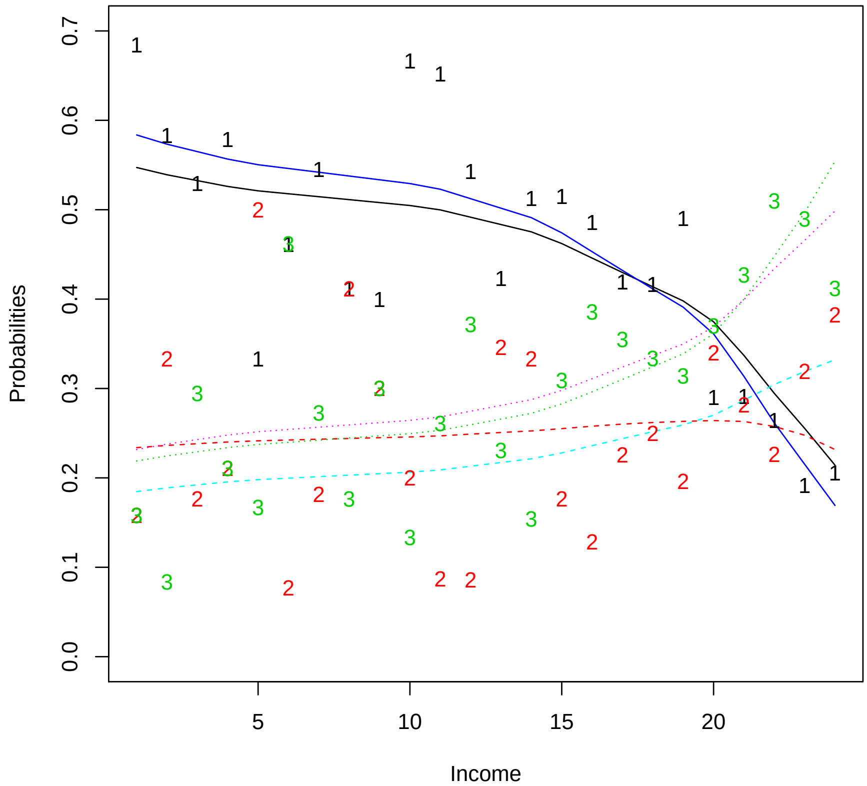
```
matplot(predict(omod2, data.frame(nincome=inca), type="probs"),
type="l", ylim=c(0, .7))
matplot(predict(mmodi, data.frame(nincome=inca),
type="probs"),
type="l", ylim=c(0, .7),add=TRUE)
matplot(prop.table(table(nincome, sPID), 1), add=TRUE)
```

3 Contingency tables

3.1 Two way tables - F4.1

Contingency tables: two-way tables

Figure 7: Ordinal fitted probs. with unordered & data



Example: Semiconductor production

	die contaminated		
wafer quality	no	yes	
good	320	14	334
bad	80	36	116
	400	50	450

Example: Framlingham heart disease study

	heart disease		
cholesterol	yes	no	
low	51	992	1043
high	41	245	286
	92	1237	1329

In each case we ask if the two factors are dependent or not?

3.2 Models - F4.1

Let y_{ij} be the number of observations with factor 1 at level i and factor 2 at level j:

	factor 2			
factor 1	1	2	3	
1	y_{11}	y_{12}	y_{13}	$y_{1\cdot}$
2	y_{21}	y_{22}	y_{23}	$y_{2\cdot}$
	$y_{\cdot 1}$	$y_{\cdot 2}$	$y_{\cdot 3}$	$y_{\cdot\cdot}$

Let π_{ij} be the probability that an observation has factor 1 at level i and factor 2 at level j.

$$
\begin{aligned}
\pi_{i\cdot} &= \textstyle\sum_j \pi_{ij} \text{ is prob. an obs. has factor 1 at level } i \\
\pi_{\cdot j} &= \textstyle\sum_i \pi_{ij} \text{ is prob. an obs. has factor 2 at level } j \\
\pi_{\cdot\cdot} &= \textstyle\sum_i \pi_{i\cdot} \;=\; \sum_j \pi_{\cdot j} \;=\; 1
\end{aligned}
$$

We want to know if $\pi_{ij} = \pi_{i\cdot}\pi_{\cdot j}$ for all i and j?

There are four different models whose application depends on the context of the data:

- Multinomial
- Poisson
- Product multinomial
- Hypergeometric

They often given the same or similar conclusions, so attention will be confined to the multinomial and product multinomial models.ß

3.3 Multinomial model - F4.1

Multinomial model

Suppose that $y_{..}$ is fixed and the model is ball throwing into the cells of the $y_{..}$ into the cells of the table, so that

$$\begin{aligned} (y_{ij})_{ij} &\sim \text{multinomial}\,(y_{..}, (\pi_{ij})_{ij}) \\ \mathbb{P}(Y_{ij} = y_{ij} \text{ for all } i \text{ and } j) &= \frac{y_{..}!}{\prod_{ij} y_{ij}!} \prod_{ij} \pi_{ij}^{y_{ij}} \end{aligned}$$

Product multinomial

Suppose that $y_{.j}$ is fixed for each j and observations are independent, then for each j

$$(Y_{ij})_i \sim \text{multinomial}(y_{.j}, (\pi_{i|j})_i)$$

where

$$\pi_{i|j} = \mathbb{P}(\text{observe factor } 1 = i \text{ given factor } 2 = j) = \frac{\pi_{ij}}{\pi_{.j}}.$$

3.4 Testing Independence- F4.1

Testing independence: multinomial model

H_0 $\pi_{ij} = \pi_{i.}\pi_{.j}$ (independent factors)

H_1 π_{ij} unrestricted.

As our test statistic we use the log likelihood ratio for the model under H_0 compared to the model under H_1.

This is just the deviance, since the model under H_1 is the full model.

Testing independence: product multinomial model

H_0 $\pi_{i|j} = \pi_{i.}$ (equivalently $\pi_{ij} = \pi_{i.}\pi_{.j}$)

H_1 λ_{ij} unrestricted.

We have a multinomial logistic regression model, where the j-th response is

$$(Y_{ij})_i \sim \text{multinomial}(y_{.j}, (p_{ji})_i)$$

and under H_0 we have that $p_{ji} = \pi_{i|j}$ does not depend on j. That is, for some β_i,

$$p_{ji} = \frac{e^{\beta_i}}{\sum_k e^{\beta_k}} \quad (= \pi_{i.})$$

Testing for H_0 is then equivalent to testing if the model with just an intercept is adequate.

Maximum Likelihood Estimators

The MLE $\hat{p_k}$ in a multinomial distribution with J categories and m balls for the probability, p_k of category k is $\hat{p_k} = y_k/m$ because the log likelihood is proportional to

$$y_1 \log(1 - \sum_{j=2}^{J} p_j) + \sum_{k=2}^{J} y_k \log(p_k).$$

And this choice of $\hat{p_k}$ is the only one which makes the partial derivative of the last expression with respect to $p_k = 0$. So for a two-table, the full model has MLE $\hat{\pi}_{ij} = y_{ij}/y_{..}$.

Maximum Likelihood Estimators - Independence

Similarly, the MLE 's for the marginal distributions are each marginal total divided by the total in the table. Thus, under the independence hypothesis, the MLE for the probability of the ij cell is the row total times the column total divided by the square of the table total. The deviance can be computed using this or through the logistic fit with only an intercept term.

3.5 Wafers example- F4.1

Data frame and table

```
y <- c(320, 14, 80, 36)
particle <- gl(2, 1, 4, labels=c("no","yes"))
quality <- gl(2, 2, 4, labels=c("good","bad"))
(wafer <- data.frame(y, particle, quality))

##     y particle quality
## 1 320       no    good
## 2  14      yes    good
## 3  80       no     bad
## 4  36      yes     bad

(ov <- xtabs(y ~ quality + particle))

##        particle
## quality  no yes
##    good 320  14
##    bad   80  36
```

Marginal proportions

```
# multinomial model
# marginal proportions for particle values
(pp <- prop.table( xtabs(y ~ particle)))

## particle
##        no       yes
## 0.8888889 0.1111111
```

```r
# marginal proportions for quality values
(qp <- prop.table( xtabs(y ~ quality)))

## quality
##      good       bad
## 0.7422222 0.2577778
```

Independence fitted values

```r
# multinomial model under independence
# # fitted values
(fv <- outer(qp,pp)*450)

##        particle
## quality       no      yes
##    good 296.8889 37.11111
##    bad  103.1111 12.88889

# deviance (on 1 d.f.)
2*sum(ov*log(ov/fv))

## [1] 54.03045

pchisq(54.03, 1, lower.tail=FALSE)

## [1] 1.974517e-13
```

Independence fitted values

So the null hypothesis of independence is very strongly rejected. An alternative is Pearson's chi-square test from MAST90105.

```r
# pearson's chisquared stat
sum((ov-fv)^2/fv)

## [1] 62.81231

summary(ov)

## Call: xtabs(formula = y ~ quality + particle)
## Number of cases in table: 450
## Number of factors: 2
## Test for independence of all factors:
##  Chisq = 62.81, df = 1, p-value = 2.274e-15
```

Via Logistic Fit

```r
# product multinomial model
(m <- matrix(y, nrow=2))
```

```
##      [,1] [,2]
## [1,]  320   80
## [2,]   14   36

modb <- glm(m ~ 1, family=binomial)
deviance(modb)

## [1] 54.03045
```