Notes for week 7

Ben Bolker

October 23, 2018


```
library(ggplot2)
theme_set(theme_bw())
library(scales)    ## squish
library(gridExtra) ## grid.arrange()
library(nnet)      ## multinom()
library(plyr)
library(reshape2)
library(faraway)   ## data
library(RColorBrewer) ## nice colours
```

Ordered predictors

(Not the primary topic but feel like I ought to mention it.)

Ordered factors are the case where there is a natural ordering to the responses.

This is (confusingly) different from the usual unordered-factor case, where the order of the levels is still used (1) to determine the order of the categories for high-level plotting and (2) to determine contrasts (which level is the baseline).

Options for dealing with ordered (or otherwise messy) predictors:

- assume linearity (equal differences in predicted values between successive levels); convert the factor back to numeric
- use `contr.sdif` from the `MASS` package
- use `ordered` instead of `factor`
- use `cut`, `cut_number`, `cut_interval` to convert continuous predictors to factors

Don't snoop!

Ordered factors: contrasts

```
ff <- function(n) {
  cc <- zapsmall(contr.poly(n))
  sign(cc)*MASS::fractions(cc^2)
```

```
}
ff(3)

##      .L   .Q
## [1,] -1/2  1/6
## [2,]    0 -2/3
## [3,]  1/2  1/6

ff(5)

##      .L    .Q    .C    ^4
## [1,]  -2/5   2/7 -1/10  1/70
## [2,] -1/10 -1/14   2/5 -8/35
## [3,]     0  -2/7     0 18/35
## [4,]  1/10 -1/14  -2/5 -8/35
## [5,]   2/5   2/7  1/10  1/70
```

No decrease in complexity, but improved interpretability. Linear, quadratic models are nested within the ordered-factor model.

Categorical responses

We can either model these as *multinomial*, or as conditional Poisson (i.e., if we take a set of independent Poisson deviates x_i they are equivalent to a multinomial sample out of $\sum_i x_i$ with $p_i = \lambda_i / \sum \lambda_i$.

In either case we have to define

$$L \propto \sum_i N_i \log p_i$$

Multinomial distributions are also conditionally *binomial* if we only want to consider one category vs. all the others ...

Here's a data set on US political preferences:

> 10 variable subset of the 1996 American National Election Study. Missing values and "don't know" responses have been listwise deleted. Respondents expressing a voting preference other than Clinton or Dole have been removed.

```
library(faraway)
data(nes96)
nn <- subset(nes96,select=c(PID,age,educ,income))
summary(nn)

##       PID           age            educ           income
##  strDem :200   Min.   :19.00   MS    : 13   $60K-$75K:103
##  weakDem:180   1st Qu.:34.00   HSdrop: 52   $50K-$60K:100
```

```
##  indDem :108   Median :44.00   HS    :248   $30K-$35K: 70
##  indind : 37   Mean   :47.04   Coll  :187   $25K-$30K: 68
##  indRep : 94   3rd Qu.:58.00   CCdeg : 90   $105Kplus: 68
##  weakRep:150   Max.   :91.00   BAdeg :227   $35K-$40K: 62
##  strRep :175                   MAdeg :127   (Other)  :473
```

For simplicity, lump party identifications into three categories:

```
nn$party <- factor(sub("(str|weak|ind)","",nn$PID))
```

Get a numeric value for the average income in a category:

```
## income breakpoints
incbrks <- c(0,3,seq(5,9,by=2),
             10:15,17,20,22,
             seq(25,50,by=5),60,75,90,105,125)
## take average of breakpoints
inca <- (incbrks[-1]+incbrks[-length(incbrks)])/2
```

Name the vector:

```
names(inca) <- levels(nn$income)
```

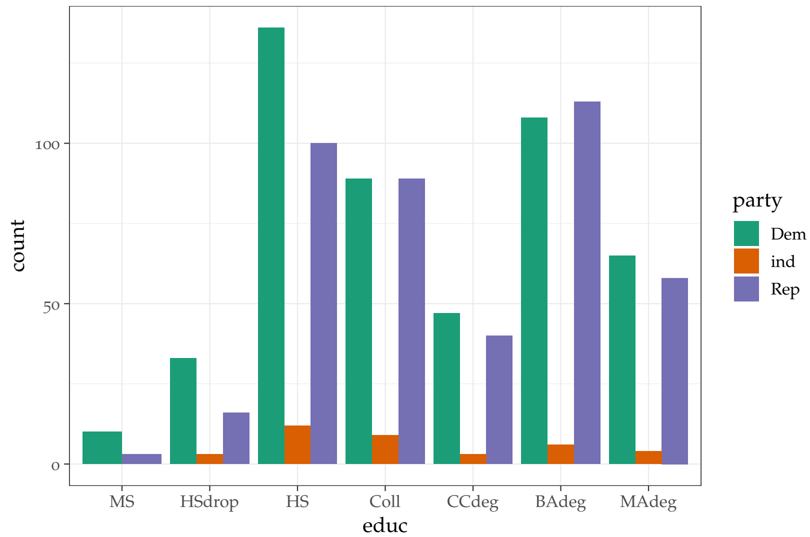
Now something like `inca["$3K-$5K"]` would work ...
Numeric versions of variables:

```
nn <- transform(nn,nincome=inca[nn$income],
                neduc=as.numeric(educ))
```

Categorical versions of variables:

```
cincome <- cut_number(nn$nincome,7)
cage <- cut_number(nn$age,7)
cdata <- with(nn,data.frame(party,educ,cincome,cage))
```

```
ggplot(cdata,aes(x=educ,fill=party))+geom_bar(position="dodge")+
    scale_fill_brewer(palette="Dark2")
```

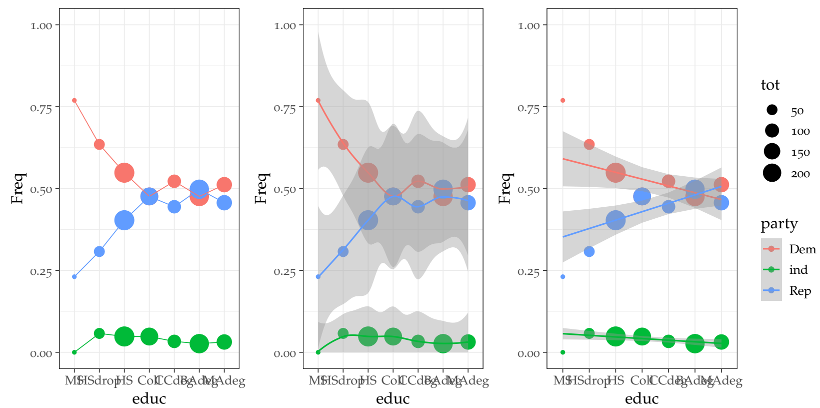


Rescale data, get proportions of parties by education and party:

```
tt <- with(nn,table(educ,party))
tot <- rowSums(tt)
tt <- sweep(tt,1,tot,"/")
tt <- data.frame(tt,tot)  ## automatically "melted"

## Warning in data.frame(tt, tot):  row names were found
from a short variable and have been discarded

tt$neduc <- as.numeric(tt$educ)
```

Three ways to plot the results:

```
g1 <- ggplot(tt,aes(x=educ,y=Freq,
                    colour=party))+
    geom_point(aes(size=tot))+
    scale_y_continuous(limits=c(0,1),oob=squish)
library(gridExtra)
g1A <- g1+geom_line(aes(group=party))+theme(legend.position="none")
g1B <- g1+geom_smooth(aes(x=as.numeric(educ)),method="loess")+
    theme(legend.position="none")
g1C <- g1 + geom_smooth(aes(group=party,weight=tot),
                        method="glm",family=binomial)

## Warning:  Ignoring unknown parameters:  family

grid.arrange(g1A,g1B,g1C,ncol=3,widths=unit(c(1,1,1.4),units="null"))
```



Multinomial responses

Non-ordered categorical responses. We have to predict the effects of *each* predictor on *each* response.

```
library(nnet)
m1 <- multinom(party ~ age+educ+nincome, nn)

## # weights:  30 (18 variable)
## initial  value 1037.090001
## iter  10 value 783.325516
## iter  20 value 756.095443
## iter  30 value 755.807940
## final  value 755.806187
## converged

summary(m1)

## Call:
## multinom(formula = party ~ age + educ + nincome, data = nn)
##
## Coefficients:
##     (Intercept)         age    educ.L    educ.Q     educ.C        educ^4
## ind   -5.136861 0.005024464 5.2444363 -6.341993 4.69342290 -2.5528042029
## Rep   -1.409234 0.010108633 0.5647774 -0.720244 0.01737136  0.0008992581
##          educ^5     educ^6    nincome
## ind  1.2918109 -0.5393994 0.01688029
## Rep -0.1032692 -0.1296745 0.01313253
##
## Std. Errors:
##     (Intercept)         age    educ.L    educ.Q    educ.C    educ^4
## ind   0.6431590 0.011118945 0.4614627 0.3962557 0.4730355 0.4716052
## Rep   0.2758803 0.004339848 0.4323886 0.3935699 0.3288897 0.2635390
```

```
##        educ^5    educ^6     nincome
## ind 0.4893813 0.4304447 0.005916620
## Rep 0.2170376 0.1762988 0.002457621
##
## Residual Deviance: 1511.612
## AIC: 1547.612
```

What do the parameters mean? e.g. the first element of the intercept vector is the log-odds of the probability of being Independent vs. Democrat in the baseline level; the second is the log-odds of the probability of being Republic vs Democrat in the baseline level.

Test this:

```
z <- data.frame(party=c("Democrat","Democrat","Ind","Republican"))
```

We take the coefficient (the intercept), compute the logistic function (`plogis`), and compute the fractional equivalent.

```
MASS::fractions(plogis(coef(multinom(party~1,data=z))))

## # weights:  6 (2 variable)
## initial  value 4.394449
## final  value 4.158883
## converged
##            (Intercept)
## Ind        1/3
## Republican 1/3
```

Both of the probabilities are 1/3 (there are 1/3 as many Independents as Democrats, and 1/3 as many Republicans as Democrats).

Change the reference level to Independent:

```
z$party <- relevel(z$party,"Ind")
```

```
MASS::fractions(plogis(coef(multinom(party~1,data=z))))

## # weights:  6 (2 variable)
## initial  value 4.394449
## final  value 4.158883
## converged
##            (Intercept)
## Democrat   2/3
## Republican 1/2
```

Compared to Independent, there are 2/3 Democrats and 1/2 Republicans . . .

Fit with numeric rather than ordinal predictors:

```
m2 <- multinom(party ~ age+neduc+nincome, nn)

## # weights:  15 (8 variable)
## initial  value 1037.090001
## iter  10 value 794.228781
## final  value 760.888806
## converged
```

Without education at all:

```
m3 <- update(m2,.~.-neduc)

## # weights:  12 (6 variable)
## initial  value 1037.090001
## iter  10 value 762.955851
## final  value 762.658537
## converged
```

What do the parameters mean??

```
summary(m2)

## Call:
## multinom(formula = party ~ age + neduc + nincome, data = nn)
##
## Coefficients:
##     (Intercept)         age       neduc    nincome
## ind   -2.560991 0.002804454 -0.21395608 0.01686278
## Rep   -1.164684 0.007441529  0.01217699 0.01302126
##
## Std. Errors:
##     (Intercept)         age      neduc     nincome
## ind   0.7862200 0.010845152 0.12194267 0.005887065
## Rep   0.3121893 0.004199209 0.04666894 0.002441064
##
## Residual Deviance: 1521.778
## AIC: 1537.778
```

To the extent that the non-intercept parameters are similar between groups, this suggests that we might be able to get away with a proportional-odds model (see below).

Finding best AIC (smallest AIC is best; $< 2\Delta\text{AIC}$ is a small difference; $> 10\Delta\text{AIC}$ is a big difference).

```
trace <- TRUE ## I don't know why, but this is necessary -- otherwise
              ## I get an error
(dd <- drop1(m1)) ## test="Chisq" is ignored

## trying - age
## # weights:  27 (16 variable)
## initial  value 1037.090001
## iter  10 value 765.518556
## iter  20 value 758.598482
## iter  30 value 758.534626
## final  value 758.534357
## converged
## trying - educ
## # weights:  12 (6 variable)
## initial  value 1037.090001
## iter  10 value 762.955851
## final  value 762.658537
## converged
## trying - nincome
## # weights:  27 (16 variable)
## initial  value 1037.090001
## iter  10 value 776.052657
## iter  20 value 772.259393
## iter  30 value 772.210493
## final  value 772.210379
## converged
##         Df      AIC
## <none>  18 1547.612
## age     16 1549.069
## educ     6 1537.317
## nincome 16 1576.421
```

Compared to best model:

```
delta_AIC <- dd$AIC-min(dd$AIC)
names(delta_AIC) <- rownames(dd)
round(delta_AIC,2)

##  <none>     age    educ nincome
##   10.30   11.75    0.00   39.10
```

We can't get p values from `drop1`, but we can do likelihood ratio tests:

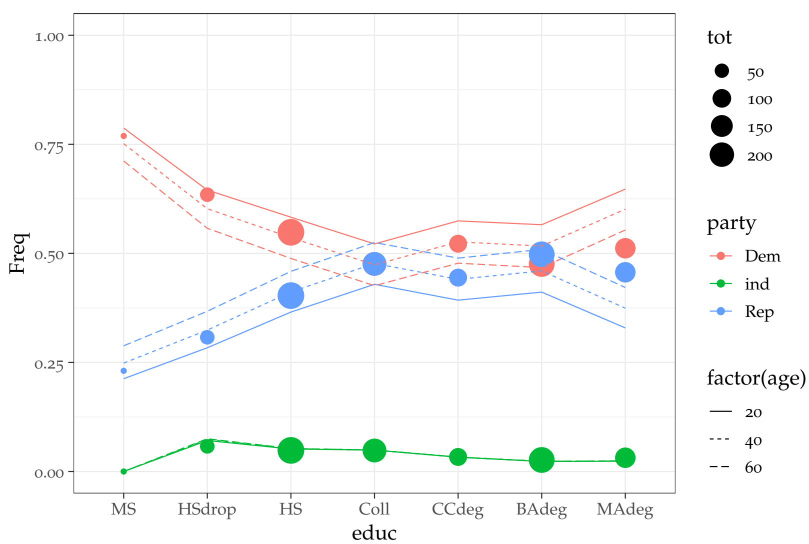
```
anova(m1,m2,m3)  ##  education: test categorical vs linear vs null model

## Likelihood ratio tests of Multinomial Models
##
## Response: party
##                     Model Resid. df Resid. Dev   Test    Df  LR stat.
## 1         age + nincome      1882   1525.317
## 2 age + neduc + nincome      1880   1521.778 1 vs 2     2  3.539461
## 3  age + educ + nincome      1870   1511.612 2 vs 3    10 10.165237
##     Pr(Chi)
## 1
## 2 0.1703789
## 3 0.4261181
```

predict.multinom ...

```
preddata <- data.frame(nincome=mean(nn$nincome),
                       expand.grid(age=c(20,40,60),educ=levels(nn$educ)))
probs <- predict(m1,newdata=preddata,type="probs")
```

```
preddata <- data.frame(preddata,probs)
predmelt <- rename(melt(preddata,id.vars=1:3),
                  c(variable="party",value="Freq"))
```

```
g1 + geom_line(aes(group=interaction(party,age),
                  lty=factor(age)),data=predmelt)
```



What else can I do with a multinomial fit?

```
methods(class="multinom")

##  [1] add1        anova       coef        confint     drop1
##  [6] extractAIC  logLik      model.frame predict     print
## [11] summary     vcov
## see '?methods' for accessing help and source code
```

(The "asterisked" functions are hidden inside the `nnet` package: e.g. to look at them you would need `nnet:::drop1.multinom`.)

Ordinal responses

Multiple categorical levels of response, but ordered.

Proportional odds (or *proportional probability*, depending on link) function).

`polr` function from the `MASS` package; also the `ordinal` package.

```
library(MASS)
p1 <- polr(party ~ age+educ+nincome, nn)
drop1(p1,test="Chisq")

## Single term deletions
##
## Model:
## party ~ age + educ + nincome
##         Df    AIC     LRT  Pr(>Chi)
## <none>     1538.8
## age      1 1542.0  5.2199   0.02233 *
## educ     6 1535.1  8.3304   0.21488
## nincome  1 1566.2 29.4579 5.715e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

p2 <- polr(party ~ age+neduc+nincome, nn)
drop1(p2,test="Chisq")

## Single term deletions
##
## Model:
## party ~ age + neduc + nincome
##         Df    AIC     LRT  Pr(>Chi)
## <none>     1537.1
## age      1 1538.0  2.9736   0.08463 .
## neduc    1 1535.1  0.0484   0.82593
## nincome  1 1564.3 29.2493 6.364e-08 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Note correlation among parameters:

```
round(cov2cor(vcov(p2)),2)

##
## Re-fitting to get Hessian

##          age neduc nincome Dem|ind ind|Rep
## age     1.00  0.13    0.03    0.74    0.74
## neduc   0.13  1.00   -0.38    0.63    0.63
## nincome 0.03 -0.38    1.00    0.12    0.12
## Dem|ind 0.74  0.63    0.12    1.00    1.00
## ind|Rep 0.74  0.63    0.12    1.00    1.00
```

Or using the `ordinal` package (more flexible/newer):

```
library(ordinal)
p3 <- clm(party ~ age+educ+nincome, data=nn)
coef(p1)

##          age       educ.L       educ.Q       educ.C       educ^4
##  0.009522628  0.573552066 -0.742893138  0.069254713 -0.044684004
##       educ^5       educ^6      nincome
## -0.081227547 -0.138260492  0.012412721

coef(p3)

##      Dem|ind      ind|Rep          age       educ.L       educ.Q
##  1.268256953  1.433490808  0.009522676  0.573548506 -0.742897303
##       educ.C       educ^4       educ^5       educ^6      nincome
##  0.069250944 -0.044695028 -0.081200313 -0.138256927  0.012412867
```

Comparing log-likelihoods and AICs between multinomial and proportional-odds models:

```
logLik(m1)

## 'log Lik.' -755.8062 (df=18)

logLik(p1)

## 'log Lik.' -759.3974 (df=10)

AIC(m1)

## [1] 1547.612
```

```
AIC(p1)

## [1] 1538.795

library(bbmle)

## Loading required package:  stats4
##
  ## Attaching package:  'bbmle'
## The following object is masked from 'package:ordinal':
  ##
  ##    slice

AICtab(m1,p1)

##    dAIC df
## p1  0.0 10
## m1  8.8 18
```