MA678 Homework 6

Jiun Lee

11/8/2022

Multinomial logit

Using the individual-level survey data from the 2000 National Election Study (data in folder NES), predict party identification (which is on a five-point scale) using ideology and demographics with an ordered multinomial logit model.

1. Summarize the parameter estimates numerically and also graphically.

```
fit_nes <- polr(partyid3 ~ ideo + race + gender + urban, Hess=TRUE, data=nes)
```

```
## Warning in polr(partyid3 ~ ideo + race + gender + urban, Hess = TRUE, data =
## nes): design appears to be rank-deficient, so dropping some coefs
```

```
summary(fit_nes)
```

```
## Call:
## polr(formula = partyid3 ~ ideo + race + gender + urban, data = nes,
##     Hess = TRUE)
##
## Coefficients:
##                                              Value Std. Error t value
## ideomoderate                                1.0170    0.05395  18.850
## ideoconservative                            1.8737    0.04737  39.555
## raceblack                                  -1.8470    0.07615 -24.257
## raceasian                                   0.3479    0.15838   2.197
## racenative american                        -0.3761    0.11110  -3.385
## racehispanic                               -0.8245    0.08099 -10.180
## genderfemale                               -0.2045    0.03715  -5.504
## urbansuburban areas                         0.4272    0.04819   8.865
## urbanrural, small towns, outlying and adja  0.2154    0.05124   4.204
##
## Intercepts:
##                           Value    Std. Error t value
## democrats|independents     1.0887   0.0556    19.5881
## independents|republicans   1.5238   0.0564    27.0200
##
## Residual Deviance: 21184.27
## AIC: 21206.27
## (26929 observations deleted due to missingness)
```

2. Explain the results from the fitted model.

```
summary(nes)
```

```
##       year          resid         weight1          weight2         weight3     
##  Min.   :1952   Min.   :   1   Min.   :0.2417   Min.   :0.00   Min.   :0.000  
##  1st Qu.:1966   1st Qu.: 504   1st Qu.:1.0000   1st Qu.:1.00   1st Qu.:1.000  
##  Median :1978   Median :1115   Median :1.0000   Median :1.00   Median :1.000  
##  Mean   :1978   Mean   :1323   Mean   :1.0561   Mean   :1.04   Mean   :1.039  
##  3rd Qu.:1990   3rd Qu.:1813   3rd Qu.:1.0000   3rd Qu.:1.00   3rd Qu.:1.000  
##  Max.   :2002   Max.   :6009   Max.   :4.0000   Max.   :4.00   Max.   :4.000  
##                                                                               
##       age           gender                   race      
##  Min.   :17.00   male  :18015   white          :33536  
##  1st Qu.:32.00   female:22094   black          : 4176  
##  Median :44.00                  asian          :  294  
##  Mean   :46.18                  native american:  609  
##  3rd Qu.:59.00                  hispanic       : 1197  
##  Max.   :99.00                  other          :  112  
##  NA's   :1267                   NA's           :  185  
##                                          educ1      
##  0. dk/ na/ no pre iw (1952)/ short-form :    0  
##  1. grade school of less (0-8 grades)    : 6103  
##  2. high school (12 grades or fewer, incl:18450  
##  3. some college(13 grades or more,but no: 7943  
##  4. college or advanced degree (no cases : 7307  
##  NA's                                    :  306  
##                                                  
##                                     urban      
##  central cities                        :10144  
##  suburban areas                        :13855  
##  rural, small towns, outlying and adja:13977  
##  NA's                                  : 2133  
##                                                
##                                                
##                                                
##                                        region     
##  0. na(1948)                              :    0  
##  1. northeast (ct,me,ma,nh,nj,ny,pa,ri,vt: 8153  
##  2. north central(il,in,ia,ks,mi,mn,mo,ne:11269  
##  3. south (al,ar,de,d.c.,fl,ga,ky,la,md,m:12771  
##  4. west (ak,az,ca,co,hi,id,mt,nv,nm,or,u: 6828  
##  NA's                                    : 1088  
##                                                 
##                                        income     
##  0. dk/ na/ refused to answer/ inap, no p:    0  
##  1. 0 to 16 percentile                   : 5624  
##  2. 17 to 33 percentile                  : 5764  
##  3. 34 to 67 percentile                  :11734  
##  4. 68 to 95 percentile                  :10759  
##  5. 96 to 100 percentile                 : 2030  
##  NA's                                    : 4198  
##                                        occup1     
##  3. skilled, semi-skilled and service wor:10528  
```

```
## 1. professional and managerial          : 8649
## 6. homemkrs(1972-92:7 in vcf0116,4 in vc: 6495
## 2. clerical and sales workers           : 6192
## 5. farmers,farm managers,farm laborers &: 1139
## (Other)                                 :  887
## NA's                                    : 6219
##                                      union
## 0. dk/na/inap, question not used (1962)/:    0
## 1. yes, someone (1948: head) in househol: 8596
## 2. no, no one in the household belongs t:30031
## NA's                                    : 1482
##
##
##
##                                     religion
## 0. dk/na/refused to answer/ no post (196:    0
## 1. protestant                           :25965
## 2. catholic (roman catholic)            : 9559
## 3. jewish                               : 1054
## 4. other and none (also includes dk pref: 3229
## NA's                                    :  302
##
##                                      educ2
## 3. 12 grades, diploma or equivalency    :9731
## 5. some college, no degree/ junior/commu:7943
## 6. ba level degrees/ advanced degrees in:7307
## 1. 8 grades or less ('grade school')    :6103
## 2. 9-12 grades ('high school'), no diplo:5526
## (Other)                                 :3193
## NA's                                    : 306
##                                      educ3
## 3. 12 grades, diploma or equivalency    :9731
## 5. some college, no degree/ junior/commu:7636
## 1. 8 grades or less ('grade school')    :6103
## 2. 9-12 grades ('high school'), no diplo:5526
## 6. ba level degree                      :5170
## (Other)                                 :5638
## NA's                                    : 305
##                                 martial_status
## 1. married and living with spouse (or sp:25139
## 2. never married                        : 4863
## 5. widowed                              : 4127
## 3. divorced                             : 3002
## 4. separated                            : 1055
## (Other)                                 :  448
## NA's                                    : 1475
##                                      occup2       icpsr_cty
## 3. skilled, semi-skilled and service wor:10938   Min.   : 1001
## 1. professional and managerial          : 8903   1st Qu.:21054
## 2. clerical and sales workers           : 6540   Median :37049
## 6. homemakers (1980-later: no other occu: 5819   Mean   :36671
## 5. farmers, farm managers, farm laborers: 1177   3rd Qu.:49221
## (Other)                                 :  919   Max.   :73037
## NA's                                    : 5813   NA's   :27071
```

```
##    fips_cty                           partyid7           partyid3
##  Min.   : 1033   2. weak democrat         :8872   democrats   :21349
##  1st Qu.:13089   1. strong democrat       :8098   independents: 3915
##  Median :27099   6. weak republican       :5949   republicans :14845
##  Mean   :28616   7. strong republican     :4985
##  3rd Qu.:42045   3. independent-democrat  :4379
##  Max.   :56013   4. independent-independent:3915
##  NA's   :19848   (Other)                  :3911
##                                        partyid3_b
##  0. dk/ na/ other/ refused to answer/ no :    0
##  1. democrats (including leaners)        :21349
##  2. indpendents and apolitical (1966 only: 3915
##  3. republicans (including leaners)      :14845
##
##
##
##                                      str_partyid
##  0. dk/ na/ other/ refused to answerl no:    0
##  1. independent or apolitical          : 3915
##  2. leaning independent                : 8290
##  3. weak partisan                      :14821
##  4. strong partisan                    :13083
##
##
##                                       father_party
##  0. na/inap/no pre iw(1952)/ no father/ n:    0
##  1. democrat                             : 8410
##  2. independent (some years also: shifter: 1085
##  3. republican                           : 4703
##  4. other/ minor party/ apolitical/ never:    0
##  9. dk (exc.1988)                        :    0
##  NA's                                    :25911
##                                       mother_party      dlikes
##  0. na/ inap/ no pre iw(1952)/ no mother/:    0   Min.   :-5.000
##  1. democrat                             : 8043   1st Qu.:-1.000
##  2. independent                          : 1165   Median : 0.000
##  3. republican                           : 4424   Mean   : 0.199
##  4. other/ minor party/ apolitical/ never:    0   3rd Qu.: 2.000
##  9. dk (excl.1988)                       :    0   Max.   : 5.000
##  NA's                                    :26477   NA's   :21292
##      rlikes         dem_therm       rep_therm         regis
##  Min.   :-5.000   Min.   : 0.00   Min.   : 0.00   Min.   :1.00
##  1st Qu.:-1.000   1st Qu.:40.00   1st Qu.:40.00   1st Qu.:2.00
##  Median : 0.000   Median :60.00   Median :60.00   Median :2.00
##  Mean   : 0.183   Mean   :57.49   Mean   :60.08   Mean   :2.72
##  3rd Qu.: 2.000   3rd Qu.:85.00   3rd Qu.:85.00   3rd Qu.:2.00
##  Max.   : 5.000   Max.   :97.00   Max.   :97.00   Max.   :9.00
##  NA's   :21292    NA's   :25859   NA's   :25801   NA's   :27143
##                                           vote
##  0. dk/na/inap, no post iw(1952,60,64...):    0
##  1. no, did not vote                     :10136
##  2. yes, voted                           :26681
##  NA's                                    : 3292
##
```

```
##
##
##                                    regisvote
##  0. dk/na if voted/dk/na if registered/in:    0
##  1. not registed, and did not vote       : 3607
##  2. registered, but did not vote         : 3825
##  3. voted (registered)                   :22342
##  NA's                                    :10335
##
##
##                                     presvote
##  0. dk/na if voted/didn't vote for pres/i:    0
##  1. democrat                             : 7130
##  2. republican                           : 7618
##  3. major third party cand (wallace 1968/:  299
##  NA's                                    :25062
##
##
##                                 presvote_2party
##  0. dk/na if voted/didn't vote for pres/i:    0
##  1. democrat                             : 7130
##  2. republican                           : 7618
##  NA's                                    :25361
##
##
##
##                                 presvote_intent    ideo_feel
##  0. dk(1964 only)/na/inap/no pre iw(1948,:    0   Min.   : 0.00
##  1. democratic candidate (with or without: 8487   1st Qu.:44.00
##  2. republican candidate (with or without: 8681   Median :49.00
##  3. undecided/ dk (exc.1964)             : 1608   Mean   :52.31
##  4. r does not intend to vote(incl. no, q:    0   3rd Qu.:59.00
##  9. other candidate                      :    0   Max.   :97.00
##  NA's                                    :21333   NA's   :14508
##                              ideo7                 ideo             cd
##  4. moderate, middle of the road: 7070   liberal     : 4455   Min.   : 1.000
##  6. conservative                : 5081   moderate    : 2985   1st Qu.: 3.000
##  5. slightly conservative       : 3819   conservative: 7569   Median : 6.000
##  2. liberal                     : 2687   NA's        :25100   Mean   : 9.057
##  3. slightly liberal            : 2435                        3rd Qu.:12.000
##  (Other)                        : 1031                        Max.   :52.000
##  NA's                           :17986                        NA's   :4152
##      state         inter_pre       inter_post       black
##  Min.   : 1.00   Min.   : 0.00   Min.   : 0.0   Min.   :0.0000
##  1st Qu.:22.00   1st Qu.:18.00   1st Qu.: 8.0   1st Qu.:0.0000
##  Median :40.00   Median :29.00   Median :17.0   Median :0.0000
##  Mean   :37.13   Mean   :29.95   Mean   :20.2   Mean   :0.1046
##  3rd Qu.:49.00   3rd Qu.:42.00   3rd Qu.:29.0   3rd Qu.:0.0000
##  Max.   :82.00   Max.   :67.00   Max.   :98.0   Max.   :1.0000
##  NA's   :1092    NA's   :18748   NA's   :5889   NA's   :185
##      female           age_sq      rep_presvote   rep_pres_intent
##  Min.   :0.0000   Min.   : 289   Min.   :0.000   Min.   :0.000
##  1st Qu.:0.0000   1st Qu.:1024   1st Qu.:0.000   1st Qu.:0.000
##  Median :1.0000   Median :1936   Median :1.000   Median :1.000
```

```
##  Mean   :0.5508   Mean   :2421   Mean   :0.517   Mean   :0.506
##  3rd Qu.:1.0000   3rd Qu.:3481   3rd Qu.:1.000   3rd Qu.:1.000
##  Max.   :1.0000   Max.   :9801   Max.   :1.000   Max.   :1.000
##                   NA's   :1267   NA's   :25361   NA's   :22941
##      south          real_ideo
##  Min.   :0.0000   Min.   :1.000
##  1st Qu.:0.0000   1st Qu.:4.000
##  Median :0.0000   Median :4.000
##  Mean   :0.2521   Mean   :4.273
##  3rd Qu.:1.0000   3rd Qu.:5.000
##  Max.   :1.0000   Max.   :7.000
##  NA's   :1092     NA's   :20651
##                                  presapprov
##  0. dk/na/inap, form ii(1972)/ question n:    0
##  1. approve                              :15002
##  2. disapprove                           : 9498
##  NA's                                    :15609
##
##
##
##                                    perfin1         perfin2
##  0. na/inap,no post(1968,72)/form ii,iii,:    0   Min.   :1.00
##  1. better now                           :10641   1st Qu.:1.00
##  2. same                                 :10602   Median :2.00
##  3. worse now                            : 8569   Mean   :1.81
##  9. dk/ uncertain/ depends               :    0   3rd Qu.:2.00
##  NA's                                    :10297   Max.   :3.00
##                                                   NA's   :34963
##      perfin         presadm           age_10        age_sq_10
##  Min.   :1.000   Min.   :-1.0000   Min.   :1.700   Min.   : 2.89
##  1st Qu.:1.000   1st Qu.:-1.0000   1st Qu.:3.200   1st Qu.:10.24
##  Median :2.000   Median : 1.0000   Median :4.400   Median :19.36
##  Mean   :1.912   Mean   : 0.1611   Mean   :4.618   Mean   :24.21
##  3rd Qu.:3.000   3rd Qu.: 1.0000   3rd Qu.:5.900   3rd Qu.:34.81
##  Max.   :3.000   Max.   : 1.0000   Max.   :9.900   Max.   :98.01
##  NA's   :5151                      NA's   :1267    NA's   :1267
##     newfathe         newmoth        parent_party        white
##  Min.   :-1.000   Min.   :-1.000   Min.   :-2.000   Min.   :0.0000
##  1st Qu.:-1.000   1st Qu.:-1.000   1st Qu.:-2.000   1st Qu.:1.0000
##  Median :-1.000   Median :-1.000   Median :-2.000   Median :1.0000
##  Mean   :-0.261   Mean   :-0.265   Mean   :-0.533   Mean   :0.8361
##  3rd Qu.: 1.000   3rd Qu.: 1.000   3rd Qu.: 2.000   3rd Qu.:1.0000
##  Max.   : 1.000   Max.   : 1.000   Max.   : 2.000   Max.   :1.0000
##  NA's   :25911    NA's   :26477    NA's   :27400
```

3. Use a binned residual plot to assess the fit of the model.

```
binnedplot(fitted(fit_nes), resid(fit_nes), xlab = "Estimated Party Identification", cex.pts=0.4)
```

```
## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA
```

```
## Warning in mean.default(y[items]): argument is not numeric or logical: returning
```

```
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
```

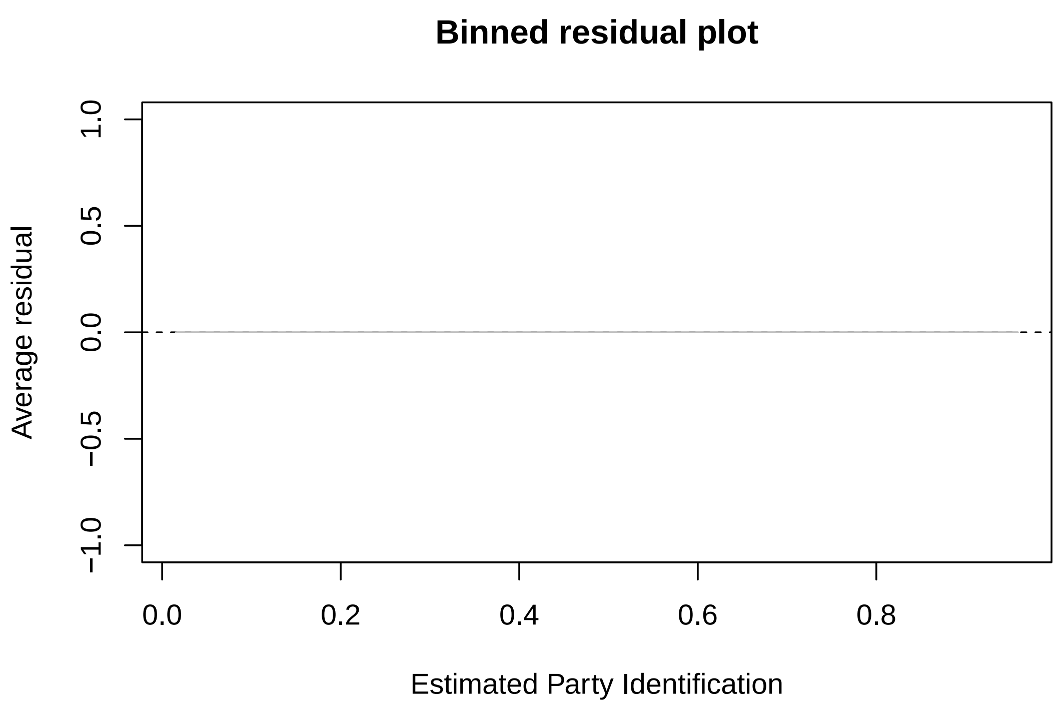
```
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
```

```
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
```

```
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
```

```
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
```

```
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA

## Warning in mean.default(y[items]): argument is not numeric or logical: returning
## NA
```



(Optional) Choice models

Using the individual-level survey data from the election example described in Section 10.9 (data available in the folder NES),

```r
nes <- read.dta("/Users/jiunlee/MSSP22/MA678/ROS-Examples-master/NES/data/nes5200_processed_voters_reali
```

1. Fit a logistic regression model for the choice of supporting Democrats or Republicans. Then interpret the output from this regression in terms of a utility/choice model.

```
#democrat:1, republican:0
index_r <- grep("republicans", nes$partyid3)
index_d <- grep("democrats", nes$partyid3)
nes <- nes[c(index_d,index_r),]
nes$partyid3 <- as.character(nes$partyid3)
nes$partyid3[which(nes$partyid3 == "1. democrats (including leaners)")] = 1
nes$partyid3[which(nes$partyid3 == "3. republicans (including leaners)")] = 0
nes$partyid3 <- as.numeric(nes$partyid3)
```

2. Repeat the previous exercise but now with three options: Democrat, no opinion, Republican. That is, fit an ordered logit model and then express it as a utility/choice mode

```
nes <- read.dta("/Users/jiunlee/MSSP22/MA678/ROS-Examples-master/NES/data/nes5200_processed_voters_real
nes$partyid3 <- as.character(nes$partyid3)
nes$partyid3[which(nes$partyid3 == "1. democrats (including leaners)")] = 1
nes$partyid3[which(is.na(nes$partyid3))] = 2
nes$partyid3[which(nes$partyid3 == "3. republicans (including leaners)")] = 3
nes$partyid3 <- as.numeric(nes$partyid3)
```

```
## Warning: NAs introduced by coercion
```

```
fit_net2 <- polr(factor(partyid3) ~ gender + race + religion + urban + income, data=nes)
```

```
## Warning in polr(factor(partyid3) ~ gender + race + religion + urban + income, :
## design appears to be rank-deficient, so dropping some coefs
```

Contingency table and ordered logit model

In a prospective study of a new living attenuated recombinant vaccine for influenza, patients were randomly allocated to two groups, one of which was given the new vaccine and the other a saline placebo. The responses were titre levels of hemaglutinin inhibiting antibody found in the blood six weeks after vaccination; they were categorized as "small", "medium" or "large".

treatment	small	moderate	large	Total
placebo	25	8	5	38
vaccine	6	18	11	35

The cell frequencies in the rows of table are constrained to add to the number of subjects in each treatment group (35 and 38 respectively). We want to know if the pattern of responses is the same for each treatment group.

1. Using a chi-square test and an appropriate log-linear model, test the hypothesis that the distribution of responses is the same for the placebo and vaccine groups.

```
chisq.test(con, correct=FALSE)
```

```
##
##  Pearson's Chi-squared test
##
## data:  con
## X-squared = 17.648, df = 2, p-value = 0.0001472
```

```
con2 <- matrix(c(25,8,5,6,18,11,0,0,0,1,1,1),nrow=6,ncol=2)
con2 <- as.data.frame(con2)
  colnames(con2) <- c("hemaglutinin","Treatment")
fit_con <- glm(log(con2$hemaglutinin)~con2$Treatment, data=con2)
```

2. For the model corresponding to the hypothesis of homogeneity of response distributions, calculate the fitted values, the Pearson and deviance residuals, and the goodness of fit statistics X^2 and D. Which of the cells of the table contribute most to X^2 and D? Explain and interpret these results.

```
library(rsq)
summary(fit_con)
```

```
##
## Call:
## glm(formula = log(con2$hemaglutinin) ~ con2$Treatment, data = con2)
##
## Deviance Residuals:
##        1         2         3         4         5         6
##  0.91629  -0.22314  -0.69315  -0.56825   0.53036   0.03789
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)     2.30259    0.40574   5.675  0.00476 **
## con2$Treatment  0.05742    0.57380   0.100  0.92510
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.4938656)
##
##     Null deviance: 1.9804  on 5  degrees of freedom
## Residual deviance: 1.9755  on 4  degrees of freedom
## AIC: 16.362
##
## Number of Fisher Scoring iterations: 2
```

```
rsq(fit_con)
```

```
## [1] 0.00249758
```

```
#The model doesn't fit well.
```

3. Re-analyze these data using ordered logit model (use `polr`) to estimate the cut-points of a latent continuous response variable and to estimate a location shift between the two treatment groups. Sketch a rough diagram to illustrate the model which forms the conceptual base for this analysis.

```
nes3 <- polr(as.factor(age) ~ income + partyid3 + religion, data=nes)
```

```
## Warning in polr(as.factor(age) ~ income + partyid3 + religion, data = nes):
## design appears to be rank-deficient, so dropping some coefs
```

High School and Beyond

The **hsb** data was collected as a subset of the High School and Beyond study conducted by the National Education Longitudinal Studies program of the National Center for Education Statistics. The variables are gender; race; socioeconomic status; school type; chosen high school program type; scores on reading, writing, math, science, and social studies. We want to determine which factors are related to the choice of the type of program—academic, vocational, or general—that the students pursue in high school. The response is multinomial with three levels.

```
data(hsb)
```

1. Fit a trinomial response model with the other relevant variables as predictors (untransformed).

```
hsb_tri <- multinom(prog ~ gender+race+ ses+ schtyp+read+write+math+science,data=hsb)
```

```
## # weights:  39 (24 variable)
## initial  value 219.722458
## iter  10 value 179.967335
## iter  20 value 157.376702
## final  value 157.146736
## converged
```

2. For the student with id 99, compute the predicted probabilities of the three possible choices.

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v tibble  3.1.8     v dplyr   1.0.9
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.2
## v purrr   0.3.4
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::between()   masks data.table::between()
## x tidyr::expand()    masks Matrix::expand()
## x dplyr::filter()    masks stats::filter()
## x dplyr::first()     masks data.table::first()
## x dplyr::lag()       masks stats::lag()
## x dplyr::last()      masks data.table::last()
## x tidyr::pack()      masks Matrix::pack()
## x dplyr::recode()    masks car::recode()
## x dplyr::select()    masks MASS::select()
## x purrr::some()      masks car::some()
## x purrr::transpose() masks data.table::transpose()
## x tidyr::unpack()    masks Matrix::unpack()
```

```r
library(dplyr)

aa <- which(hsb$id == 99)
id99 <- hsb %>% slice(aa, preserve=FALSE)
predict(hsb_tri, newdata = id99, "probs")
```

```
##  academic   general  vocation
## 0.4614919 0.3629782 0.1755298
```

Happiness

Data were collected from 39 students in a University of Chicago MBA class and may be found in the dataset `happy`.

```r
library(faraway)
data(happy)
```

1. Build a model for the level of happiness as a function of the other variables.

```r
happy$happy = factor(happy$happy, levels = c("1","2","3","4","5","6","7","8","9","10"), ordered = TRUE)
happy$love = factor(happy$love, levels = c("1", "2","3"), ordered = TRUE)
happy$sex = factor(happy$sex, levels = c("0", "1"), ordered = TRUE)
happy$work = factor(happy$work, levels = c("1", "2","3","4","5"), ordered = TRUE)

library(MASS)
fit_happy <- polr(happy ~ money+sex+love+work, data= happy)
```

2. Interpret the parameters of your chosen model.

```r
exp(coef(fit_happy))
```

```
##       money       sex.L      love.L      love.Q      work.L      work.Q
##   1.0179821   0.4847672 256.8749933   1.4672709   3.9422187   1.1763273
##      work.C      work^4
##   0.1801833   0.7977230
```

```r
#For money, when family income increases in 1 thousand dollars, the odds of 2point~10point combined ver

#One unit increase in sex, from 0 (not satisfactory) to 1 (satisfactory sex activity), the odds of 5poi

#For love, when love=1(lonely), the odds of 10 points versus 1point~9points combined increase by 256.87

#For work,  when work=1, the odds of 8points~10points combined versus 1point~7points combined increase
```

3. Predict the happiness distribution for subject whose parents earn $30,000 a year, who is lonely, not sexually active and has no job.

```r
#money = 30, love=1, sex=0, work=1
pred_happy <- expand.grid(money=30,love="1",sex="0",work="1")
predict(fit_happy,newdata=pred_happy,type = "probs")
```

```
##            1            2            3            4            5            6
## 4.538644e-04 2.017327e-01 1.697647e-01 4.971082e-01 1.118276e-01 8.461385e-03
##            7            8            9           10
## 9.605046e-03 1.025472e-03 1.736889e-05 3.551740e-06
```

Newspaper survey on Vietnam War

A student newspaper conducted a survey of student opinions about the Vietnam War in May 1967. Responses were classified by sex, year in the program and one of four opinions. The survey was voluntary. The data may be found in the dataset `uncviet`. Treat the opinion as the response and the sex and year as predictors. Build a proportional odds model, giving an interpretation to the estimates.

```r
library(MASS)
library(rstanarm)
```

```
## Loading required package: Rcpp

## This is rstanarm version 2.21.3

## - See https://mc-stan.org/rstanarm/articles/priors for changes to default priors!

## - Default priors may change, so it's safest to specify priors, even if equivalent to the defaults.

## - For execution on a local, multicore CPU with excess RAM we recommend calling

##   options(mc.cores = parallel::detectCores())

##
## Attaching package: 'rstanarm'

## The following objects are masked from 'package:VGAM':
##
##     cauchy, dirichlet, exponential, laplace, logit

## The following object is masked from 'package:faraway':
##
##     logit

## The following object is masked from 'package:car':
##
##     logit

## The following objects are masked from 'package:arm':
##
##     invlogit, logit
```

```r
data(uncviet)

pom <- stan_polr(factor(uncviet$policy) ~ uncviet$year + uncviet$year, prior=R2(0.3, "mean"))
```

```
## Warning: Omitting the 'data' argument is not recommended and may not be allowed
## in future versions of rstanarm. Some post-estimation functions (in particular
## 'update', 'loo', 'kfold') are not guaranteed to work properly unless 'data' is
## specified as a data frame.
```

```
##
## SAMPLING FOR MODEL 'polr' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 8.3e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.83 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.313153 seconds (Warm-up)
## Chain 1:                0.27745 seconds (Sampling)
## Chain 1:                0.590603 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'polr' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 1.9e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.19 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
```

```
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.299181 seconds (Warm-up)
## Chain 2:                0.242764 seconds (Sampling)
## Chain 2:                0.541945 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'polr' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 1.8e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.18 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.284895 seconds (Warm-up)
## Chain 3:                0.335857 seconds (Sampling)
## Chain 3:                0.620752 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'polr' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 1.6e-05 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.16 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.311105 seconds (Warm-up)
## Chain 4:                0.294463 seconds (Sampling)
```

```
## Chain 4:                0.605568 seconds (Total)
## Chain 4:
```

```
summary(pom)
```

```
##
## Model Info:
##  function:     stan_polr
##  family:       ordered [logistic]
##  formula:      factor(uncviet$policy) ~ uncviet$year + uncviet$year
##  algorithm:    sampling
##  sample:       4000 (posterior sample size)
##  priors:       see help('prior_summary')
##  observations: 40
##
## Estimates:
##                       mean   sd   10%   50%   90%
## uncviet$yearGrad    0.0    0.6 -0.8   0.0   0.8
## uncviet$yearJunior  0.0    0.6 -0.8   0.0   0.8
## uncviet$yearSenior  0.0    0.6 -0.8   0.0   0.8
## uncviet$yearSoph    0.0    0.6 -0.8   0.0   0.8
## A|B               -1.1    0.5 -1.8  -1.1  -0.5
## B|C                0.0    0.5 -0.6   0.0   0.6
## C|D                1.2    0.5  0.5   1.2   1.9
##
## Fit Diagnostics:
##               mean   sd   10%   50%   90%
## mean_PPD:A 0.3    0.1  0.1   0.2   0.4
## mean_PPD:B 0.2    0.1  0.1   0.2   0.4
## mean_PPD:C 0.2    0.1  0.1   0.2   0.4
## mean_PPD:D 0.3    0.1  0.1   0.2   0.4
##
## The mean_ppd is the sample average posterior predictive distribution of the outcome variable (for de
##
## MCMC diagnostics
##                     mcse Rhat n_eff
## uncviet$yearGrad   0.0  1.0  3927
## uncviet$yearJunior 0.0  1.0  3850
## uncviet$yearSenior 0.0  1.0  3906
## uncviet$yearSoph   0.0  1.0  3933
## A|B                0.0  1.0  4230
## B|C                0.0  1.0  4142
## C|D                0.0  1.0  3981
## mean_PPD:A         0.0  1.0  4003
## mean_PPD:B         0.0  1.0  3835
## mean_PPD:C         0.0  1.0  3764
## mean_PPD:D         0.0  1.0  4311
## log-posterior      0.0  1.0  1707
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of effective sample
```

Pneumonoconiosis of coal miners

The pneumo data gives the number of coal miners classified by radiological examination into one of three categories of pneumonoconiosis and by the number of years spent working at the coal face divided into eight categories.

```
data(pneumo, package = "faraway")
```

1. Treating the pneumonoconiosis status as response variable as nominal, build a model for predicting the frequency of the three outcomes in terms of length of service and use it to predict the outcome for a miner with 25 years of service.

```
pneumo.reg <- multinom(status ~ year, data = pneumo)
```

```
## # weights:  9 (4 variable)
## initial  value 26.366695
## final  value 26.366695
## converged
```

```
summary(pneumo.reg)
```

```
## Call:
## multinom(formula = status ~ year, data = pneumo)
##
## Coefficients:
##         (Intercept)         year
## normal 2.109424e-15 2.486900e-14
## severe 2.664535e-15 3.552714e-14
##
## Std. Errors:
##        (Intercept)       year
## normal    1.142515 0.03420049
## severe    1.142515 0.03420049
##
## Residual Deviance: 52.73339
## AIC: 60.73339
```

```
predict(pneumo.reg, data.frame (year = 25), type = "probs")
```

```
##      mild    normal    severe
## 0.3333333 0.3333333 0.3333333
```

2. Repeat the analysis with the pneumonoconiosis status being treated as ordinal.

```
pneumo.order <- polr(factor(status) ~ year, data = pneumo, Hess = TRUE)
summary(pneumo.order)
```

```
## Call:
## polr(formula = factor(status) ~ year, data = pneumo, Hess = TRUE)
##
```

```
## Coefficients:
##          Value Std. Error   t value
## year 4.341e-11    0.02565 1.692e-09
##
## Intercepts:
##               Value   Std. Error t value
## mild|normal   -0.6931  0.8838    -0.7842
## normal|severe  0.6931  0.8838     0.7842
##
## Residual Deviance: 52.73339
## AIC: 58.73339
```

```
predict(pneumo.order, data.frame (year = 25), type = "probs")
```

```
##      mild    normal    severe
## 0.3333333 0.3333333 0.3333333
```

3. Now treat the response variable as hierarchical with top level indicating whether the miner has the disease and the second level indicating, given they have the disease, whether they have a moderate or severe case.

```
pneumo$status.h <- ifelse(pneumo$status == "normal", 0, 1)

pneumo.sub <- as.data.frame(cbind(Freq = pneumo$Freq, normal = ifelse(pneumo$status == "normal",1,0), mi

pneumo.hie <- multinom(cbind(normal,mild,severe) ~ year, data = pneumo.sub)
```

```
## # weights:  9 (4 variable)
## initial  value 26.366695
## final  value 26.366695
## converged
```

```
summary(pneumo.hie)
```

```
## Call:
## multinom(formula = cbind(normal, mild, severe) ~ year, data = pneumo.sub)
##
## Coefficients:
##         (Intercept)         year
## mild   1.221245e-15 2.486900e-14
## severe 2.664535e-15 3.552714e-14
##
## Std. Errors:
##        (Intercept)       year
## mild      1.142515 0.03420049
## severe    1.142515 0.03420049
##
## Residual Deviance: 52.73339
## AIC: 60.73339
```

```r
predict(pneumo.hie, data.frame(year = 25), type = "probs")
```

```
##    normal      mild    severe
## 0.3333333 0.3333333 0.3333333
```

4. Compare the three analyses.

```r
#The results of three analyses are not so different, but they are with mild around 0.08-0.10, normal ar
```