Logit example with 2002 ANES data

Paul Johnson, CRMDA, pauljohn@ku.edu

Keywords: single-authoring, just one
See https://crmda.ku.edu/guides for updates.

Oct. 4, 2018

Abstract

Some variables from the American National Election Study of 2002 are used to demonstrate logistic regression in R. The same old SAS export file that we created 2003 still works, all these years later.

Contents

[1]This is the Ian Ostrander memorial section

Address line 1
Address line 2
City State Zipcode

Web: https://crmda.ku.edu
Email: you@where.edu
Phone: 123-345-5678

1 Get that ANES 2002 dataset

In 2003, some students and I used SAS to pull variables from the American National Election Study. That effort was immortalized in a post on the KU server, then called lark. When lark closed, I copied that material onto my personal web server, http://pj.freefaculty.org/DataSets/ANES/2002. The result of the importation was a SAS export file, which I called "PJTEST.sasxport" because I did not think it would work. The code (shown below) will download that file and rename it more confidently, "anes2002.sasxport."

The data import process today might be handled differently, but this one is still to be cherished because of its special place in time. That was the semester when I quit teaching stats with SAS and asked the students to use R (R Core Team (2018)).

There is a SAS export dataset that is the primary source material for this exercise.

There is one very important thing to explain about this data. The variable labels were lost, we have just the numeric values for them. Hence, to create factors in R, we have to do a little work.

Now, in 2018, I'd be telling everybody "don't bother with that, use the variable key in the kutils package." But I realize that's an unfamiliar thing to most people, so I'll do the recoding the old fashioned way here.

1.1 Check the Codebook

The full variable list of the original survey is online:

http://pj.freefaculty.org/DataSets/ANES/2002/RELEASE/nes02var.txt

The data set itself is retrieved in R code below.

Some of the highlights in the codebook are as follows.

V023131 Y3x. Summary: R Education

What is the highest grade of school or year of college you completed?

1. 8 grades or less and no diploma or equivalency
2. 9-11 grades, no further schooling
3. High school diploma or equivalency test
4. More than 12 years of schooling, no higher degree
5. Junior or community college level degrees (AA degrees)
6. BA level degrees; 17+ years, no advanced degree
7. Advanced degree, including LLB
9. Refused

0. NA in Y3, Y3a or Y3b

V023131 Frequency ————————-

0 2

1 36

2 70

3 399

4 313

5 155

6 347

7 178

9 11

==========

V023111 Q1a. Who did R vote for in 2000 Pres Election

Which one did you vote for?

1. Al Gore

3. George W. Bush

5. Ralph Nader

7. Other {SPECIFY}

8. Dont know

9. Refused 0. NA

INAP. Fresh Cross-Section respondent; 5,8,9 in Q1

V023111 Frequency ————————- .

526 0 1 1 431 3 502 5 32 7 8 8 1 9 10

==========

V023036 J1. R Consider Self Dem/Rep/Ind Numeric Missing eq 0, ge 8

J1.

Generally speaking, do you usually think of yourself as a REPUBLICAN, a DEMOCRAT, an INDEPENDENT, or what? ————————————

1. Democrat

2. Republican

3. Independent

4. Other Party {VOL} {SPECIFY}

5. No Preference {VOL}

8. Don't know

9. Refused 0. NA

V023036 Frequency ————————-

0 13

1 502

2 474

3 429

4 27

5 59

8 5

9 2

==========

V023022 R 7Pt Scale Lib-Con Self-Placement

We hear a lot of talk these days about liberals and conservatives. When it comes to politics, do you usually think of yourself as EXTREMELY LIBERAL, LIBERAL, SLIGHTLY LIBERAL, MODERATE OR MIDDLE OF THE ROAD, SLIGHTLY CONSERVATIVE, CONSERVATIVE, EXTREMELY CONSERVATIVE, or haven't you thought much about this? ————————————————

01. Extremely Liberal

02. Liberal

03. Slightly Liberal

04. Moderate; Middle of the Road

05. Slightly Conservative

06. Conservative

07. Extremely Conservative

08. Don't know

09. Refused

90. Haven't thought much [Do Not Probe]

00. NA

V023022 Frequency ————————-

0 11

1 23

2 181

3 135

4 340

5 186

6 315

7 65

8 8

9 3

90 244

==========

V023027 H1. US Economy Better/Worse in Last Yr

Now thinking about the economy in the country as a whole, would you say that over the past year the nation's economy has gotten BETTER, STAYED ABOUT THE SAME, or gotten WORSE?

———

1. Better

3. Same

5. Worse

8. Don't know

9. Refused

0. NA

V023027 Frequency ————————-

0 6

1 69

3 321

5 1112

8 3

==========

V023010: Where on that thermometer would you rate George W. Bush? (on a scale from 0 to 100)

==========

1.2 Use foreign::read.xport to import the data

R's package "foreign" has a number of procedures to import data from other packages. Happily, this data imports correctly.

First, I'll retrieve a copy of the file from the web server. I'll rename it "anes2002.sasxport". To keep this simple, I'll just drop it in the current working directory. (In real projects, I'd put it in a separate folder.)

```
url <-
   "http://pj.freefaculty.org/DataSets/ANES/2002/PJTEST.sasxport"
fn <- "anes2002.sasxport"
if(!file.exists(fn)){
     download.file(url, mode="wb", destfile="anes2002.sasxport")
 }
```

The SAS export file is easy to import

```
library(foreign)
nes2002 <- read.xport(fn)
```

I've got 1511 rows and 732 columns.

Unfortunately, all of the variable labels were lost in the process, I have only the numeric scores. I found that out by using the str() function:

```
str(nes2002)
```

```
'data.frame': 1511 obs. of  732 variables:
 $ DSETID  : Factor w/ 1 level "NES_DATASET:2002.T": 1 1 1 1 1 1 1 1 1 1 ...
 $ VERSION : Factor w/ 1 level "VERSION:20030807": 1 1 1 1 1 1 1 1 1 1 ...
 $ ICPSRNO : num  3740 3740 3740 3740 3740 3740 3740 3740 3740 3740 ...
 $ V020001 : num  1 2 3 4 5 6 7 8 9 10 ...
 $ V020002 : num  943 347 341 912 1106 ...
 $ V020101 : num  0.956 1.466 0.5 0.352 0.498 ...
 $ V020102 : num  0 1.295 0.46 0 0.489 ...
 $ V021001 : num  1 1 1 1 1 1 1 1 1 1 ...
 $ V021002 : num  1 2 2 1 2 2 2 1 1 2 ...
 $ V021100 : num  NA NA NA NA NA NA NA NA NA NA ...
 $ V021101A: num  NA NA NA NA NA NA NA NA NA NA ...
 $ V021101B: num  NA NA NA NA NA NA NA NA NA NA ...
 $ V021101C: num  NA NA NA NA NA NA NA NA NA NA ...
 $ V021101D: num  NA NA NA NA NA NA NA NA NA NA ...
 $ V021102A: num  NA NA NA NA NA NA NA NA NA NA ...
 $ V021102B: num  NA NA NA NA NA NA NA NA NA NA ...
 $ V021102C: num  NA NA NA NA NA NA NA NA NA NA ...
 $ V021102D: num  NA NA NA NA NA NA NA NA NA NA ...
 $ V021102E: num  NA NA NA NA NA NA NA NA NA NA ...
 $ V021103A: num  NA NA NA NA NA NA NA NA NA NA ...
 $ V021103B: num  NA NA NA NA NA NA NA NA NA NA ...
 $ V021103C: num  NA NA NA NA NA NA NA NA NA NA ...
 $ V021103D: num  NA NA NA NA NA NA NA NA NA NA ...
 $ V021103E: num  NA NA NA NA NA NA NA NA NA NA ...
 $ V021104A: num  NA NA NA NA NA NA NA NA NA NA ...
 $ V021104B: num  NA NA NA NA NA NA NA NA NA NA ...
 $ V021104C: num  NA NA NA NA NA NA NA NA NA NA ...
 $ V021104D: num  NA NA NA NA NA NA NA NA NA NA ...
 $ V021104E: num  NA NA NA NA NA NA NA NA NA NA ...
 $ V021105 : num  NA NA NA NA NA NA NA NA NA NA ...
 $ V021106 : num  NA NA NA NA NA NA NA NA NA NA ...
 $ V021107 : num  NA NA NA NA NA NA NA NA NA NA ...
 $ V021107A: num  NA NA NA NA NA NA NA NA NA NA ...
 $ V021107B: num  NA NA NA NA NA NA NA NA NA NA ...
 $ V021107C: num  NA NA NA NA NA NA NA NA NA NA ...
 $ V021107D: num  NA NA NA NA NA NA NA NA NA NA ...
 $ V021107E: num  NA NA NA NA NA NA NA NA NA NA ...
 $ V021107F: num  NA NA NA NA NA NA NA NA NA NA ...
 $ V021108 : num  NA NA NA NA NA NA NA NA NA NA ...
 $ V021109 : num  NA NA NA NA NA NA NA NA NA NA ...
 $ V021201 : num  6 48 39 49 53 19 42 39 48 49 ...
```

```
$ V021201A: num  71 49 24 67 73 31 14 24 49 67 ...
$ V021201B: Factor w/ 48 levels "AL","AR","AZ",..: 4 41 33 42 45 11 36 33 41 42 ...
$ V021202 : num  4 22 5 2 9 5 3 11 8 2 ...
$ V021202A: num  604 4822 3905 4902 5309 ...
$ V021202B: num  7104 4922 2405 6702 7309 ...
$ V021202C: Factor w/ 390 levels "9999","AL02",..: 21 347 268 357 378 110 294 274 333 357
   ...
$ V021203 : num  4 3 2 4 4 2 1 2 3 4 ...
$ V021204 : num  1 1 1 1 1 2 2 1 2 1 ...
$ V021205 : num  NA NA NA NA NA NA NA NA NA NA ...
$ V021206 : num  NA NA NA NA NA NA NA NA NA NA ...
$ V021207 : num  NA NA NA NA NA NA NA NA NA NA ...
$ V021208 : num  NA NA NA NA NA NA NA NA NA NA ...
$ V022000 : num  3 4 2 7 8 4 8 2 8 5 ...
$ V022001 : num  3 2 3 3 2 3 3 2 3 3 ...
$ V022002 : num  1 1 1 1 1 1 1 1 1 1 ...
$ V022003 : num  4 9 25 14 2 4 17 18 15 6 ...
$ V022004 : num  1 1 1 1 1 1 1 1 1 1 ...
$ V022005 : num  1 1 1 1 1 1 1 1 1 1 ...
$ V022006 : num  5 5 5 5 5 5 5 5 5 5 ...
$ V022007 : num  5 5 5 5 5 5 5 5 5 5 ...
$ V022008 : num  5 5 5 5 5 5 5 5 5 5 ...
$ V022009 : num  5 5 5 5 5 5 5 5 5 5 ...
$ V022010 : num  NA NA NA NA NA NA NA NA NA NA ...
$ V022011 : num  1 1 1 1 1 1 1 1 1 1 ...
$ V022012A: num  11 11 11 11 10 10 11 11 11 10 ...
$ V022012B: num  4 3 3 2 26 31 3 1 3 28 ...
$ V022012C: Factor w/ 48 levels "0918","0919",..: 48 47 47 46 39 44 47 45 47 41 ...
$ V022012D: num  1 2 2 3 10 5 2 4 2 8 ...
$ V022013A: num  11 11 11 11 10 10 11 11 11 10 ...
$ V022013B: num  4 3 3 2 26 31 3 1 3 28 ...
$ V022013C: Factor w/ 48 levels "0918","0919",..: 48 47 47 46 39 44 47 45 47 41 ...
$ V022014 : num  1 1 1 1 1 1 1 1 1 1 ...
$ V022015 : num  1 1 1 1 1 1 1 1 1 1 ...
$ V022016 : num  50 50 50 50 20 50 50 50 50 50 ...
$ V022017A: num  11 11 11 11 11 11 11 11 11 11 ...
$ V022017B: num  7 7 7 7 4 5 7 7 7 5 ...
$ V022018 : num  NA 37 40 50 25 29 36 30 43 29 ...
$ V022019 : num  16 3 2 4 6 3 1 12 14 16 ...
$ V022020 : num  0 0 0 0 0 0 0 0 0 0 ...
$ V022021 : Factor w/ 7 levels "","0930","1010",..: 1 1 1 1 1 1 1 1 1 1 ...
$ V022022 : num  NA NA NA NA NA NA NA NA NA NA ...
$ V022023 : num  1 1 1 1 0 1 1 1 1 1 ...
$ V022024 : num  154 160 200 75 59 55 180 20 76 94 ...
$ V022025 : num  2 2 2 NA 2 1 2 NA 2 1 ...
$ V022026 : num  5 5 4 NA 4 4 4 NA 3 4 ...
$ V022027 : num  1 NA 1 NA 1 1 2 NA 1 1 ...
$ V022027A: num  0 1 0 NA 0 0 0 NA 0 0 ...
$ V022028 : num  1 0 1 NA 0 0 0 NA 0 0 ...
$ V022029 : num  1 1 1 NA 0 1 0 NA 2 0 ...
$ V022030 : num  1 1 1 NA 1 1 1 NA 1 4 ...
$ V022401 : num  1 2 2 1 2 2 2 1 2 1 ...
$ V022402 : num  2 1 1 2 2 2 1 2 1 1 ...
$ V022403 : num  1 1 2 2 2 2 2 2 2 2 ...
$ V022404 : num  2 2 2 1 2 2 2 1 1 2 ...
$ V022405 : num  1 1 1 2 2 1 2 1 2 2 ...
$ V022406 : num  1 1 1 NA NA 2 NA 1 NA NA ...
$ V022407 : num  1 1 1 NA NA 1 NA 2 NA NA ...
$ V022408 : num  1 1 1 NA NA 1 NA 1 NA NA ...
 [list output truncated]
```

1.3 Recodes

I've got a few challenges. For categorical variables, it is best to use R factors, rather than integers to record the information. This would have been easier if I had made a variable key, I'm sorry I did not. But I'll make an informal variable table to help trace this through.

Table 1: Variable Coding Table (not a real variable key, but similar)

name_old	name_new		class_new	value_new	
V023111	bushvotef		factor	"Gore", "Bush"	
V023111	bushvote		integer	0=Gore,1=Bush	
V023036	partyid		factor	"D","R","I"	
V023036	repub		integer	1="R",0="D","I",or"NA"	
V023036	democ		integer	1="D",0="R","I",or"NA"	
V023022	ideolf		factor	"EL", "L",...,"EC"	
V023022	ideolo		ordered	"EL", "L",...,"EC"	
V023022	ideoln		integer	1-7	
V023010	V023010		numeric		Bush Thermometer, unchanged
V023027	useconn		integer		
V023027	usecono		ordered	1="Better", 3="Same",5="Worse"	
V023131	educn		numeric	1-7	
V023131	educf		factor	"elem.","Some HS","HS","Some coll.","AA","BA","Grad"	unordered coding
V023131	educo		ordered	same	ordinal coding

1. Create "bushvote", a binary variable for Gore vs Bush, and all of the other candidates are treated as missing.

```
nes2002$bushvotef <- factor(nes2002$V023111, levels = c(1, 3),
   labels = c("Gore", "Bush"))
table(nes2002$bushvote, nes2002$V023111, exclude = NULL)
```

```
        1   3   5   7 <NA>
 Gore 431   0   0   0    0
 Bush   0 502   0   0    0
 <NA>   0   0  32   8  538
```

```
nes2002$bushvote <- ifelse(nes2002$bushvotef == "Bush", 1, 0)
table(nes2002$bushvote, nes2002$V023111)
```

```
    1   3   5   7
 0 431   0   0   0
 1   0 502   0   0
```

2. For political party, it looks like I featured the simple 3 category variable, V023036, not the 7 point "strength of identification" scale that you see in some projects. First I'll assign labels for the 3 most widely used party labels.

```
nes2002$partyid <- factor(nes2002$V023036, levels = c(3,1, 2),
   labels = c("I", "D", "R"))
table(nes2002$partyid, nes2002$V023036, exclude = NULL)
```

```
        1   2   3   4   5 <NA>
 I      0   0 429   0   0    0
 D    502   0   0   0   0    0
 R      0 474   0   0   0    0
 <NA>   0   0   0  27  59   20
```

In the analysis below, I have some illustrations using democrat and republican dummy variables.

```
nes2002$democ <- ifelse(nes2002$partyid == "D", 1, 0)
table(nes2002$democ, nes2002$partyid, exclude = NULL)
```

```
        I   D   R <NA>
 0    429   0 474    0
 1      0 502   0    0
 <NA>   0   0   0  106
```

```
nes2002$repub <-ifelse(nes2002$partyid == "R", 1, 0)
table(nes2002$repub, nes2002$partyid, exclude = NULL, dnn =
   list("repub", "partyid"))
```

```
     partyid
repub    I   D   R <NA>
  0    429 502   0    0
  1      0   0 474    0
  <NA>   0   0   0  106
```

As I look at that now, I have some misgivings about it because of the way missing values are handled. Can you see why I might worry?

(Concern is that when we say democ = 0, we might mean everybody who is not a democrat, EVEN including missing values on party. In this current coding, we don't have that, we have missings on democ unless party is D, R, or I.

The most I think about that, the more I think I ought to fix it. So here's a fix:

```
nes2002$democ <- 0
nes2002$democ[!is.na(nes2002$partyid) & nes2002$partyid ==
   "D"] <- 1
table(nes2002$democ, nes2002$partyid, exclude = NULL)
```

```
     I   D   R <NA>
 0 429   0 474  106
 1   0 502   0    0
```

```
nes2002$repub <- 0
nes2002$repub[!is.na(nes2002$partyid) & nes2002$partyid ==
   "R"] <- 1
table(nes2002$repub, nes2002$partyid, exclude = NULL, dnn =
   list("repub", "partyid"))
```

```
     partyid
repub   I   D   R <NA>
    0 429 502   0  106
    1   0   0 474    0
```

The important thing about this "fix" is that we mean to say that democ is 1 if a person is a democrat, and they are 0 no matter what otherwise. They are 0 even if they did not answer the party identification question. This is aggressive coding, I think. It is the way we always used to do it in SAS, however.

3. Fix Ideology. Some authors treat ideology as a numeric variable, as if 1 through 7 are numerically meaningful scores. Some authors say "that's a categorical variable, don't do that." We can try both ways. Hence we have `ideolf` and `ideoln`. The variable `ideolo` is an ordinal factor. Ordinal factors are treated in an interestingly different way in the regression model (will demonstrate).

```
nes2002$ideolf <- factor(nes2002$V023022, levels = 1:7, labels
   = c("EL", "L", "SL", "M", "SC", "C", "EC"))
table("ideology factor" = nes2002$ideolf, "original ideology"
   = nes2002$V023022, exclude = NULL)
```

```
                original ideology
ideology factor   1   2   3   4   5   6   7 <NA>
          EL     23   0   0   0   0   0   0    0
          L       0 181   0   0   0   0   0    0
          SL      0   0 135   0   0   0   0    0
          M       0   0   0 340   0   0   0    0
          SC      0   0   0   0 186   0   0    0
          C       0   0   0   0   0 315   0    0
          EC      0   0   0   0   0   0  65    0
          <NA>    0   0   0   0   0   0   0  266
```

```
nes2002$ideolo <- factor(nes2002$V023022, levels = 1:7, labels
   = c("EL", "L", "SL", "M", "SC", "C", "EC"), ordered = TRUE)
```

I will also make a numeric version by converting the factor back to numeric

```
nes2002$ideoln <- as.numeric(nes2002$ideolf) # converts to 1
   thru 7
table(nes2002$ideolf, nes2002$ideoln, dnn = list("factor
   version", "ideology: numeric version"), exclude = NULL)
```

```
              ideology: numeric version
factor version   1   2   3   4   5   6   7 <NA>
          EL    23   0   0   0   0   0   0    0
          L      0 181   0   0   0   0   0    0
          SL     0   0 135   0   0   0   0    0
          M      0   0   0 340   0   0   0    0
          SC     0   0   0   0 186   0   0    0
          C      0   0   0   0   0 315   0    0
          EC     0   0   0   0   0   0  65    0
          <NA>   0   0   0   0   0   0   0  266
```

In the discussion below, I compare an ideology variable coded 1-2-3-4-5-6-7 against one that is coded with labels.

4. Education

```
## Categorical, unordered
nes2002$educf <- factor(nes2002$V023131, levels = 1:7,
          labels = c("Elem.","Some HS","HS","Some
             coll.","AA","BA","Grad"))
## Ordered categories
nes2002$educo <- factor(nes2002$V023131, levels = 1:7,
          labels = c("Elem.","Some HS","HS","Some
             coll.","AA","BA","Grad"),
          ordered = TRUE)
## Numeric
nes2002$educn <- nes2002$V023131
table("Education(ordered)" = nes2002$educo,
       "Education (orig)" = nes2002$V023131, exclude = NULL)
```

```
                 Education (orig)
Education(ordered)   1   2   3   4   5   6   7 <NA>
       Elem.        36   0   0   0   0   0   0    0
       Some HS       0  70   0   0   0   0   0    0
       HS            0   0 399   0   0   0   0    0
       Some coll.    0   0   0 313   0   0   0    0
       AA            0   0   0   0 155   0   0    0
       BA            0   0   0   0   0 347   0    0
       Grad          0   0   0   0   0   0 178    0
       <NA>          0   0   0   0   0   0   0   13
```

5. US Economy

```
nes2002$usecono <- factor(nes2002$V023027, levels = c(1, 3, 5),
          labels = c("Better", "Same", "Worse"),
          ordered = TRUE)
nes2002$useconn <- nes2002$V023027
table("US Econ(ordered)" = nes2002$usecono,
       "US Econ (orig)" = nes2002$V023027, exclude = NULL)
```

```
               US Econ (orig)
US Econ(ordered)    1    3    5 <NA>
         Better    69    0    0    0
         Same       0  321    0    0
         Worse      0    0 1112    0
         <NA>       0    0    0    9
```

2 Ideology and the vote

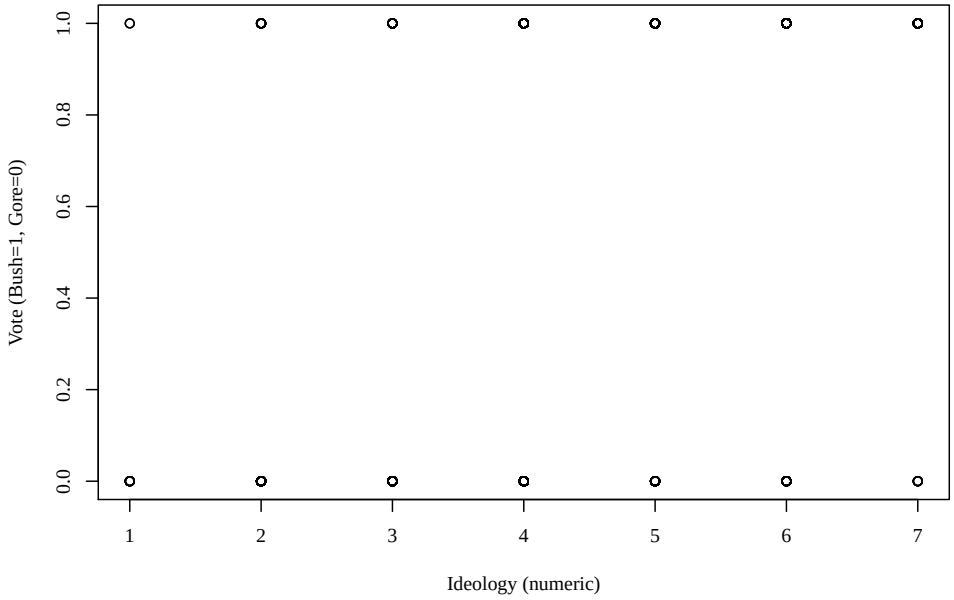
I decided to "be stupid" and simply treat Ideology as a 7 point numerical variable (even though it is actually an ordered factor).

Plot The Vote's Dependence on Ideology

Consider the very uninformative plot in Figure 1. That's really quite pathetic. The visual impact is not quite so depressing in Figure 2, which uses "jittered" values to display overlapped points. That figure includes the predicted values from the OLS regression (discussed next).

Figure 1: Scatterplot showing impact of ideology on vote choice in 2000

```
plot(nes2002$ideoln, nes2002$bushvote, ylim=c(0,1), xlab="Ideology
   (numeric)", ylab="Vote (Bush=1, Gore=0)")
```



Ordinary Least Squares

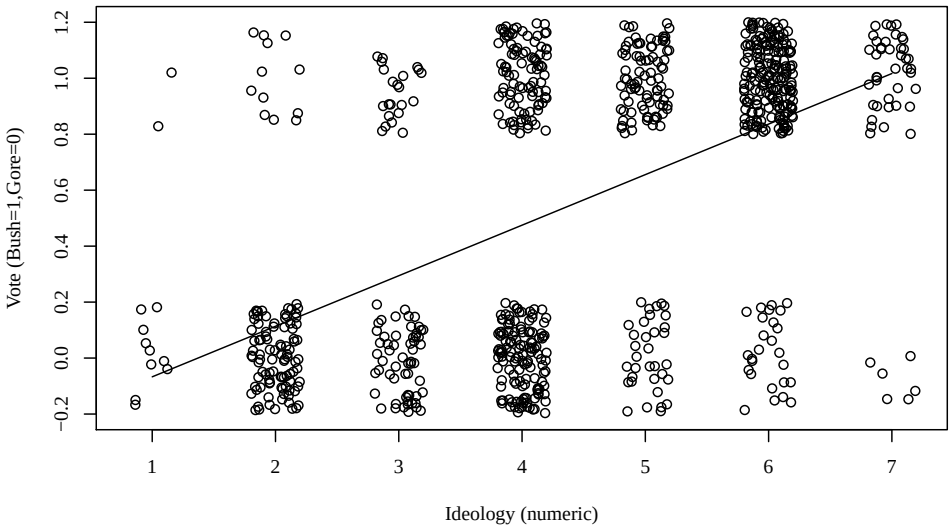
The ordinary least squares result is obtained with the `lm` (short for "linear model") function in R. The fitted model is saved as an object that we call ols1. The summary results are obtained by applying the summary function to the result.

```
ols1 <- lm(bushvote ~ ideoln, data=nes2002)
summary(ols1)
```

```
Call:
lm(formula = bushvote ~ ideoln, data = nes2002)

Residuals:
    Min      1Q  Median      3Q     Max
-1.0167 -0.2943  0.1639  0.3445  1.0670

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.247610   0.045310  -5.465 6.18e-08 ***
ideoln       0.180620   0.009624  18.768  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4149 on 807 degrees of freedom
  (702 observations deleted due to missingness)
Multiple R-squared:  0.3038,  Adjusted R-squared:  0.303
F-statistic: 352.2 on 1 and 807 DF,  p-value: < 2.2e-16
```

Figure 2: Do jittered values reveal more information?

```
with(nes2002, plot(jitter(ideoln), jitter(bushvote), ylim=c(-0.2,
   1.2),
    xlab="Ideology (numeric)", ylab="Vote (Bush=1,Gore=0)"))
nd <- data.frame(ideoln = 1:7)
nd$fit <- predict(ols1, newdata = nd)
lines(nd$ideoln, nd$fit)
```



Logistic estimation with glm

The R team prefers that we think of the logit model as a Generalized Linear Model (GLM). We are supposed to think of the outcomes as draws from a Binomial distribution (coin flips) with probability equal to the following formula:

$$P(Y_i = 1) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X_i)}}$$

Imagine searching through the possible combinations of $\hat{\beta}_0$ and $\hat{\beta}_1$. For each combination, we can calculate $\hat{p}_i$, an estimate of the probability that the i'th observation will be 1.

$$\hat{p}_i = \frac{1}{1 + e^{-(\hat{\beta}_0 + \hat{\beta}_1 X_i)}}$$

The estimator will tune $\hat{\beta}_0$ and $\hat{\beta}_1$ up and down, trying to make the observed data as likely as possible.

The "raw" output from R is as follows.

```
bushideoglm1 <- glm(bushvote ~ ideoln, family=binomial,
   data=nes2002)
```

```
summary(bushideoglm1)
```

```
Call:
glm(formula = bushvote ~ ideoln, family = binomial, data = nes2002)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.3568  -0.7596   0.5632   0.8583   2.4691

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -3.95168    0.31728  -12.46   <2e-16 ***
ideoln       0.95209    0.07001   13.60   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1110.80  on 808  degrees of freedom
Residual deviance:  833.91  on 807  degrees of freedom
  (702 observations deleted due to missingness)
AIC: 837.91

Number of Fisher Scoring iterations: 4
```

Presentable Tables

The first package I know of that incorporated a high-quality LaTeX table was "memisc", authored by Martin Elff. The regression table has some interesting features. The LaTeX rendering from the use of the memisc package is displayed in Table 2:

Table 2: Logistic Regression Table (memisc)

```
toLatex(mtable(bushideoglm1))
```

(Intercept)	-3.952^{***}
	(0.317)
ideoln	0.952^{***}
	(0.070)
Aldrich-Nelson R-sq.	0.255
McFadden R-sq.	0.249
Cox-Snell R-sq.	0.290
Nagelkerke R-sq.	0.388
phi	1.000
Likelihood-ratio	276.885
p	0.000
Log-likelihood	-416.956
Deviance	833.913
AIC	837.913
BIC	847.304
N	809

The strength of this is the profuse listing of summary statistics. That is also the weakness.

I don't want student papers to include a profusion of summary statistics that they don't understand. That's why rockchalk::outreg's output, which is displayed in the bottom part of Table 3.

In outreg, there is a parameter to specify variable names. I will accumulate the new names here in a vector named vl, and then put it to use when necessary.

```
vl = c("ideoln" = "Ideology", "ideolf" = "Ideology",
       "ideolo" = "Ideology", "repub" = "Repub.",
       "democ" = "Democ.", "V023010" = "Bush Therm",
       "useconn" = "US economy", "usecono" = "US economy",
       "educn" = "Education", educo = "Education", educf =
         "Education",
       "partyidD" = "Party Dem.", "partyidR" = "Party Rep.",
       "ideolo^4" = "ideolo.4", "ideolo^5" = "ideolo.5",
       "ideolo^6" = "ideolo.6")
```

Table 3: Logistic Regression Table(outreg)

```
outreg(bushideoglm1, tight=F, varLab = vl)
```

	M1	
	Estimate	(S.E.)
(Intercept)	-3.952***	(0.317)
Ideology	0.952***	(0.070)
N	809	
Deviance	833.913	
$-2LLR(Model\chi^2)$	276.885***	

$*p \leq 0.05** p \leq 0.01***p \leq 0.001$

2.1 How to interpret the slope estimates.

Most of our emphasis is on the estimates of the β's. The strategy is to calculate predictions and compare across cases. By doing this, we find out the answer to the question, "does the predictor effect the chances that the outcome will be 1?"

The probability is defined in this two step procedure.

1. Calculate the Linear Predictor

$$\hat{\eta}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i + more\,\hat{\beta} \cdot x's \tag{1}$$

2. "Bend" the Linear Predictor into a Probability curve

$$\hat{p}_i = Prob(Y_i = 1|\hat{\eta}_i) = \frac{1}{1 + e^{-\hat{\eta}}} \tag{2}$$

The intercept and the predictor variables all "blend together" in the linear predictor to control the chances that the outcome will be 1.

The best way to understand the β's is to calculate $\hat{p}_i$ for "interesting" values of the input variables.

Please remember this very important thing:

To calculate a predicted value, one must include values for all input variables and all parameters.

That means the "effect of x" does really depend on the values of all of the other input variables.

The usual approach is to set all of the variables at their "means" or "modes" and then adjust the values of a particular input to findout how the predicted probability changes. In the rockchalk package, I set the categorical variables at their modes, unless the user directs otherwise.

2.2 Use the predict function

In the past, I have calculated predicted values "manually" by taking the estimated coefficients and using them in the logistic equation. R offers a function "predict" to handle that for us. This is a fundamental skill that all R users should try to master.

In this case, since ideology takes on values from 1 to 7, we need only calculate 7 predictions. To obtain predicted values from the `glm` output, one uses the following approach. First, a "newdata" object, called `nd`, is created. Second, the model is given to the predict method, along with the newdata object:

```
nd <- data.frame(ideoln = 1:7)
pp3 <-predict(bushideoglm1, newdata=nd, type="response", se.fit=T)
pp3
```

```
$fit
        1         2         3         4         5         6         7
0.04744458 0.11430586 0.25060247 0.46423380 0.69185001 0.85331917 0.93778746

$se.fit
        1         2         3         4         5         6         7
0.01132256 0.01885170 0.02403794 0.02210266 0.02046537 0.01784950 0.01185117

$residual.scale
[1] 1
```

Unlike models fitted with `lm`, the `predict` function for `glm` models does not create either a confidence or prediction interval. The calculation of the confidence intervals for those predictions is still a controversial topic. There are at least 20 suggested methods, and I only know how to do the calculations for the simplest one. The approach I take is the familiar "plus or minus 2 standard errors" approach, dubbed the Wald confidence interval. The interval is calculated in the un-transformed "log odds" scale as $(\hat{\eta}_i - 1.96 se.fit(\hat{\eta}_i), \hat{\eta}_i + 1.96 se.fit(\hat{\eta}_i))$. Those values are mapped onto the scale of the predicted probabilities in the usual way, $1/(1 + exp(-\eta_i)$. Example output is shown in Figure 3.

rockchalk to make that automatic? Dependable enough

Because some students have trouble putting together the sequence of calculations, the `rockchalk` package has functions that are intended to do this work. These are described in the vignette `rockchalk`, but here's the high level overview.

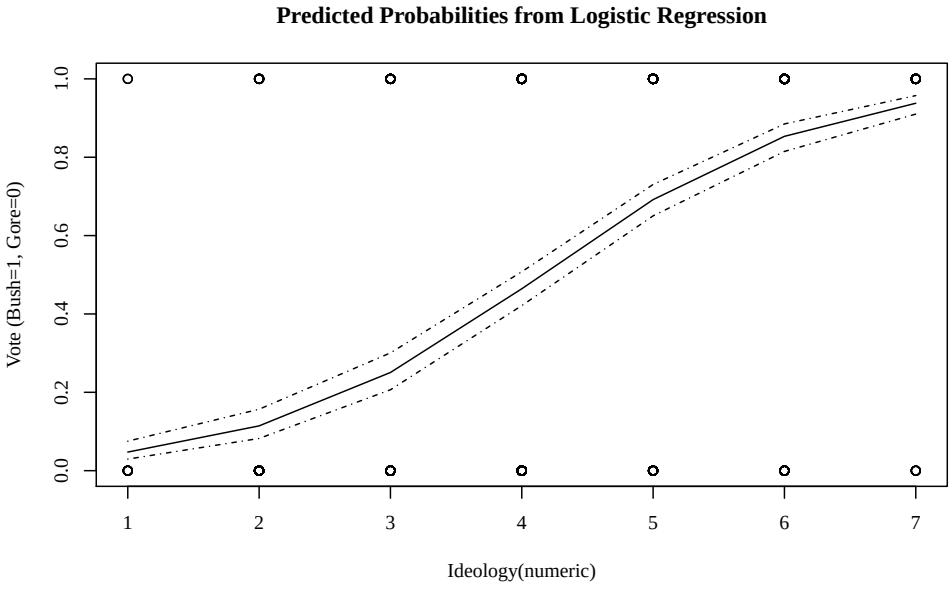
The `predictOMatic` function is supposed to look at the regression, find the predictors, get "interesting" example values, and then calculate the predictions (possibly with a confidence interval). It

Figure 3: Fitted Values and Confidence Interval of Logistic Regression

```
plot(bushvote ~ ideoln, data=nes2002, ylim=c(0,1),
   xlab="Ideology(numeric)", ylab="Vote (Bush=1, Gore=0)",
   main="Predicted Probabilities from Logistic Regression")
pp3 <- predict(bushideoglm1, newdata=nd, se.fit=T)
pp3$upr <- pp3$fit + 1.96*pp3$se.fit
pp3$lwr <- pp3$fit - 1.96*pp3$se.fit
pp3logistic <- lapply(pp3, plogis)
pp3logistic
```

```
$fit
        1         2         3         4         5         6         7
0.04744458 0.11430586 0.25060247 0.46423380 0.69185001 0.85331917 0.93778746

$se.fit
       1        2        3        4        5        6        7
0.5623081 0.5464180 0.5319556 0.5222017 0.5239802 0.5355915 0.5506091

$residual.scale
[1] 0.7310586

$upr
       1        2        3        4        5        6        7
0.0752618 0.1567613 0.3005821 0.5077160 0.7304548 0.8849724 0.9573485

$lwr
        1         2         3         4         5         6         7
0.02957990 0.08222736 0.20647990 0.42128863 0.65036267 0.81477864 0.91009764
```

```
lines(1:7, pp3logistic$fit)
lines(1:7, pp3logistic$lwr, lty=4)
lines(1:7, pp3logistic$upr, lty=4)
```



turns out that deciding what is "interesting" is subjective, so the predVals parameter has a *whole lot* of options. At the current time, the options are difficult to understand and it is necessary to study the help page and the examples to truly grasp the power and beauty of this fully operational death star. Here I take the easy road by simply listing the values for the one predictor.

```
predictOMatic(bushideoglm1, predVals = list(ideoln=1:7),
   interval="confidence")
```

```
rockchalk:::predCI: model's predict method does not return an interval.
 We will improvize with a Wald type approximation to the confidence interval
  ideoln        fit        lwr        upr
1      1 0.04744458 0.02958016 0.07526117
2      2 0.11430586 0.08222787 0.15676037
3      3 0.25060247 0.20648066 0.30058110
4      4 0.46423380 0.42128941 0.50771522
5      5 0.69185001 0.65036346 0.73045414
6      6 0.85331917 0.81477941 0.88497190
7      7 0.93778746 0.91009824 0.95734824
```

I think it is useful to run `predictOMatic`, especially if an output table of predicted probabilities is needed. These can be used to create a predicted value plot.

On the other hand, it is not necessary for the user to run `predictOMatic` to get a nice looking plot. The `plotCurves` function will run `predictOMatic` in the background and draw a plot. An example of `plotCurves` is found in Figure 4.

```
plotCurves(bushideoglm1, plotx=ideoln, interval="confidence", ylim
   = c(0, 1.2))
```

```
rockchalk:::predCI: model's predict method does not return an interval.
 We will improvize with a Wald type approximation to the confidence interval
```

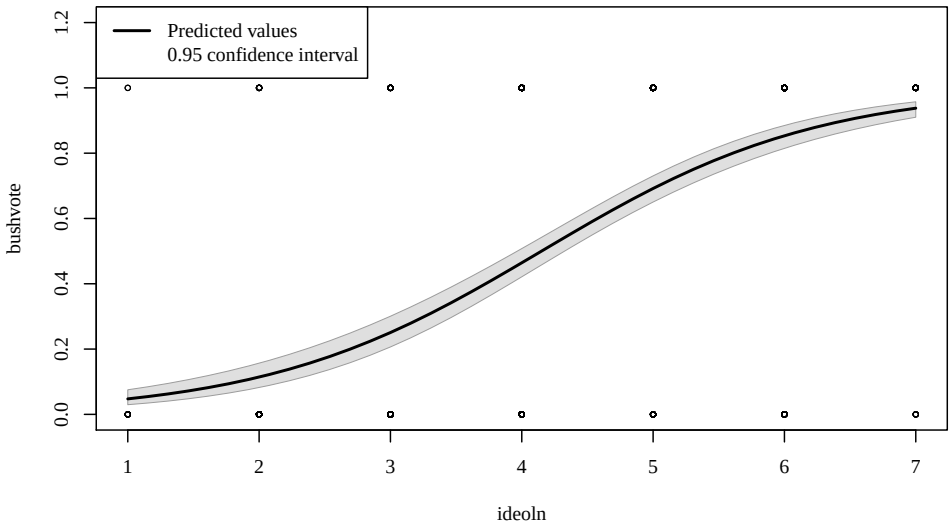


Figure 4: plotCurves demonstration with ideology

predictOMatic might fail, plotCurves might fail. These fail when the regression model being estimated has some unusual characteristic, something I did not plan for.

In which case I'd fall back to the two step procedure.

1. make a "newdata" data frame that has one row for each hypothetical case for which I want to make a prediction, and
2. run predict (as shown in Figure 3) and then fiddle around with the fit and se.fit values to manufacture the confidence intervals.

Because making the newdata object can be frustrating, I created a function called newdata in rockchalk .

3 Put party into the logistic regression

In 2003, I include this section only because I thought Figure 5 was really neat. I believe this shows that the effect of ideology is easier to spot if you contrast Democrats and Republicans. Now, in 2018, I have a better way to estimate the model and make the plot.

But I think it is somewhat unfair to students if I give you the awesome result we have at the current time without admitting that I blundered about while I was finding the better way. So I'll include the old demonstration, then the new way.

The 2003 way I included party id in the model

The party variable was recoded into two "dummy" variables, "repub" id 1 for people who are in the Republican party, 0 for others, and "democ", represents membership in the Democratic party.

```
#start adding variables
bushideoglm2 <- glm(bushvote ~ ideoln + repub + democ,
   family=binomial, data=nes2002)
summary(bushideoglm2)
```

```
Call:
glm(formula = bushvote ~ ideoln + repub + democ, family = binomial,
    data = nes2002)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.8323  -0.4131   0.2644   0.4982   2.9662

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -2.38246    0.37840  -6.296 3.05e-10 ***
ideoln       0.65621    0.08672   7.567 3.82e-14 ***
repub        1.78155    0.28679   6.212 5.23e-10 ***
democ       -2.66062    0.27573  -9.649  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1110.80  on 808  degrees of freedom
```

```
Residual deviance:  553.48  on 805  degrees of freedom
  (702 observations deleted due to missingness)
AIC: 561.48

Number of Fisher Scoring iterations: 5
```

To estimate the impact of ideology for Democrats and Republicans, I went through an inconvenient process of making up 3 separate new data objects, and pulling predictions for each one. Today, I'd make one newdata object here, but I have to live with the silliness of the past here:

```
lnth <- length(nes2002$ideoln)
newdfD1 <-data.frame(ideoln=seq(1,7), repub=0, democ=1)
newdfD1
```

```
  ideoln repub democ
1      1     0     1
2      2     0     1
3      3     0     1
4      4     0     1
5      5     0     1
6      6     0     1
7      7     0     1
```

```
predvalsD1 <- predict(bushideoglm2, newdata=newdfD1,
   type="response")
newdfR1 <-data.frame(ideoln=seq(1,7), repub=1, democ=0)
newdfR1
```

```
  ideoln repub democ
1      1     1     0
2      2     1     0
3      3     1     0
4      4     1     0
5      5     1     0
6      6     1     0
7      7     1     0
```

```
predvalsR1 <- predict(bushideoglm2, newdata=newdfR1,
   type="response")
newdfI1 <-data.frame(ideoln=seq(1,7), repub=0, democ=0)
predvalsI1 <- predict(bushideoglm2, newdata=newdfI1,
   type="response")
```

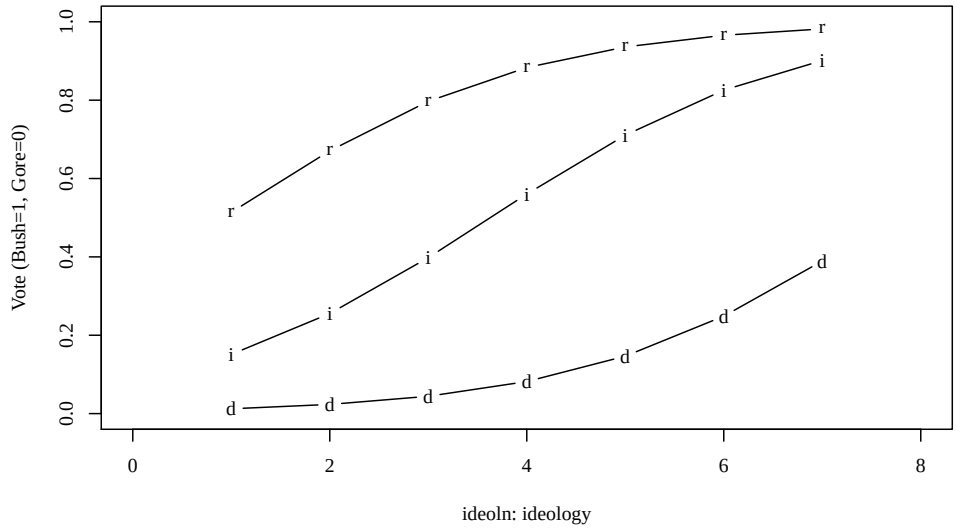
Making a better party & ideology model

There was no reason to use the dummies `repub` and `democ`. I have a factor variable `partyid` and I might as well use it. The additive model is:

```
bushideoglm3 <- glm(bushvote ~ ideoln + partyid, family=binomial,
   data=nes2002)
summary(bushideoglm3)
```

Figure 5: Democrats and Republicans

```
plot(seq(1,7), predvalsD1, type="n", ylim=c(0,1), xlim=c(0,8),
   xlab="ideoln: ideology", ylab="Vote (Bush=1, Gore=0)")
text(seq(1,7), predvalsR1, labels=rep("r",7))
text(seq(1,7), predvalsD1, labels=rep("d",7))
text(seq(1,7), predvalsI1, labels=rep("i",7))
lines(seq(1,7), predvalsR1, type="c")
lines(seq(1,7), predvalsD1, type="c")
lines(seq(1,7), predvalsI1, type="c")
```



```
Call:
glm(formula = bushvote ~ ideoln + partyid, family = binomial,
    data = nes2002)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.8430  -0.4116   0.2621   0.3634   2.9820

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -2.52894    0.39468  -6.408 1.48e-10 ***
ideoln       0.66957    0.08977   7.459 8.73e-14 ***
partyidD    -2.57512    0.28083  -9.170  < 2e-16 ***
partyidR     1.86562    0.29120   6.407 1.49e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1074.69  on 780  degrees of freedom
Residual deviance:  519.68  on 777  degrees of freedom
  (730 observations deleted due to missingness)
AIC: 527.68

Number of Fisher Scoring iterations: 5
```

The jury might be undecided if this model is actually better. Here is the main problem. The missing values in `partyid` cause some cases to be lost. The

```
predictOMatic(bushideoglm3, predVals = list(ideoln=1:7, partyid =
    c("I", "D", "R")))
```

```
   ideoln partyid        fit
1       1       I 0.13477628
2       2       I 0.23329473
3       3       I 0.37279895
4       4       I 0.53726680
5       5       I 0.69400623
6       6       I 0.81585125
7       7       I 0.89641958
8       1       D 0.01172205
9       2       D 0.02264480
10      3       D 0.04329955
11      4       D 0.08122814
12      5       D 0.14726640
13      6       D 0.25225303
14      7       D 0.39722049
15      1       R 0.50156082
16      2       R 0.66280408
17      3       R 0.79337450
18      4       R 0.88235879
19      5       R 0.93610768
20      6       R 0.96623895
21      7       R 0.98242723
```

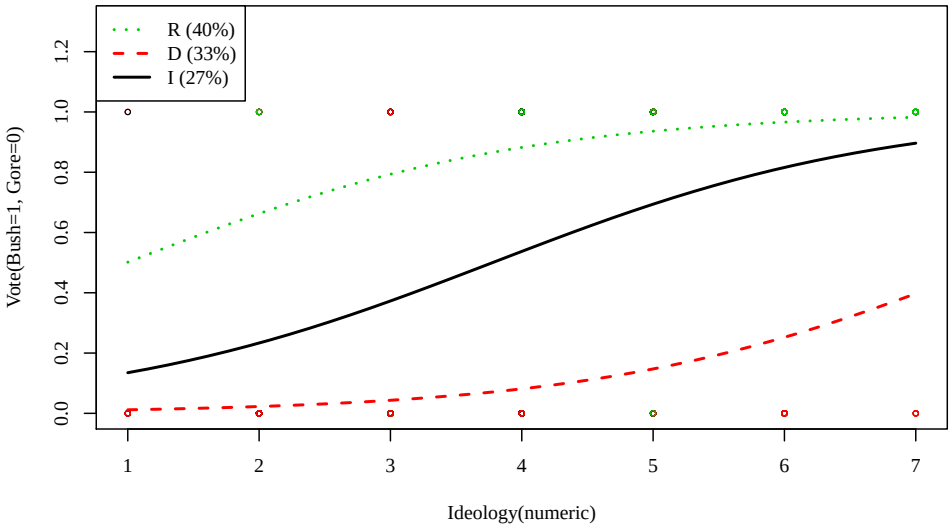
Should it be treated as an interaction model?

We assumed party id was an additive element, but perhaps it is interactive.

There was no reason to use the dummies repub and democ. I have a factor variable partyid and I might as well use it. The additive model is:

Figure 6: plotCurves for ideology with moderator party identification

```
plotCurves(bushideoglm3, plotx=ideoln, modx=partyid, ylim = c(0,
   1.3),
              xlab = "Ideology(numeric)", ylab = "Vote(Bush=1,
                 Gore=0)")
```



```
bushideoglm4 <- glm(bushvote ~ ideoln * partyid, family=binomial,
    data=nes2002)
summary(bushideoglm4)
```

```
Call:
glm(formula = bushvote ~ ideoln * partyid, family = binomial,
    data = nes2002)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.8414  -0.4345   0.2624   0.3635   2.5001

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)      -3.9410     0.6768  -5.823 5.79e-09 ***
ideoln            1.0171     0.1619   6.283 3.31e-10 ***
partyidD          0.6046     0.9363   0.646 0.518440
partyidR          3.2869     1.1228   2.927 0.003419 **
ideoln:partyidD  -0.7612     0.2233  -3.409 0.000651 ***
ideoln:partyidR  -0.3495     0.2457  -1.423 0.154873
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1074.69  on 780  degrees of freedom
Residual deviance:  507.24  on 775  degrees of freedom
  (730 observations deleted due to missingness)
AIC: 519.24

Number of Fisher Scoring iterations: 6
```

Interactions for free: Problem of the S curve translation

One problem I have not emphasized much in class is that all glm have an inherent "interaction" effect, even if we don't estimate models with interaction terms. Even if you don't run $y_i = \beta_0 + \beta_1 x1_i + \beta_2 x2_i + \beta_3 x1_i x2_i$, many generalized models behave *as if* you did that.

In logistic regression, that's easy to see because the S-shaped curve bends everything. Party ID is important, so much so that it appears that ideology has not much effect on Republicans or Democrats in Figure 7. In order to "see" the interaction, apart from the S-shaped transformation, we can inspect the model in the space of the linear predictor.

The `predictOMatic` output includes predictions of the η_i variable, the linear predictor. The default `predictOMatic` output included the transformed values, $(\frac{1}{1+e^{-X\beta}})$.

```
predictOMatic(bushideoglm3, predVals = list(ideoln=1:7, partyid =
    c("I", "D", "R")), type = "link")
```

```
  ideoln partyid         fit
1      1       I -1.85937191
2      2       I -1.18979989
3      3       I -0.52022786
4      4       I  0.14934416
5      5       I  0.81891619
6      6       I  1.48848822
7      7       I  2.15806024
8      1       D -4.43449212
```

```
9       2       D -3.76492009
10      3       D -3.09534807
11      4       D -2.42577604
12      5       D -1.75620402
13      6       D -1.08663199
14      7       D -0.41705996
15      1       R  0.00624331
16      2       R  0.67581534
17      3       R  1.34538736
18      4       R  2.01495939
19      5       R  2.68453142
20      6       R  3.35410344
21      7       R  4.02367547
```

See Figure 8.

4 The Bush Thermometer variable is interesting.

This is introduced because it is *as close to a numeric variable as we can get* in this data. I believe it is actually an ordinal variable masquerading as a number, but we'll treat it as a number, just for practice.

The variable in question is

- V023010: Where on that thermometer would you rate George W. Bush? (on a scale from 0 to 100)

That one does not need to be recoded. Its histogram is displayed in Figure 9.

The OLS estimates for the model including only the Bush feeling thermometer are as follows:

```
olsthermo1 <- lm(bushvote ~ V023010, data=nes2002)
summary(olsthermo1)
```

```
Call:
lm(formula = bushvote ~ V023010, data = nes2002)

Residuals:
    Min      1Q  Median      3Q     Max
-0.9629 -0.3299  0.1004  0.2903  1.2779

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.3031905  0.0327025  -9.271   <2e-16 ***
V023010      0.0126614  0.0004567  27.726   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3691 on 929 degrees of freedom
  (580 observations deleted due to missingness)
Multiple R-squared:  0.4528,  Adjusted R-squared:  0.4522
F-statistic: 768.7 on 1 and 929 DF,  p-value: < 2.2e-16
```

The results from logistic estimates from glm are as follows (see Figure 10).

```
glmthermo1 <- glm(bushvote ~ V023010, data=nes2002,
   family=binomial)
summary(glmthermo1)
```

Figure 7: plotCurves for ideology with interaction

```
plotCurves(bushideoglm4, plotx=ideoln, modx=partyid, ylim = c(0,
   1.3), xlab = "Ideology(numeric)", ylab = "Vote(Bush=1, Gore=0)")
```

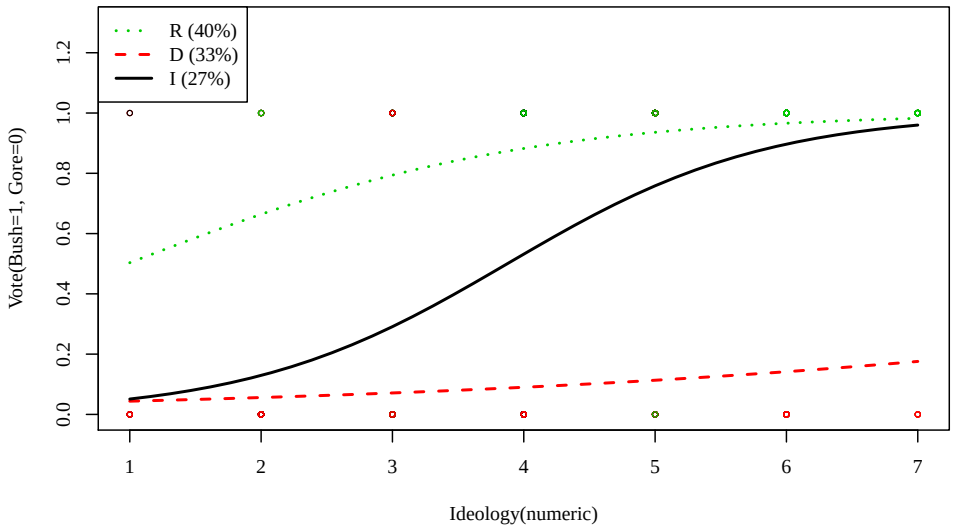


Figure 8: plotCurves in the linear predictor space to better inspect interactions

```
plotCurves(bushideoglm4, plotx=ideoln, modx=partyid,
             xlab="Ideology(numeric)", ylab="Linear predictor: XB",
             type="link", plotPoints=FALSE, ylim = c(-4,4))
```

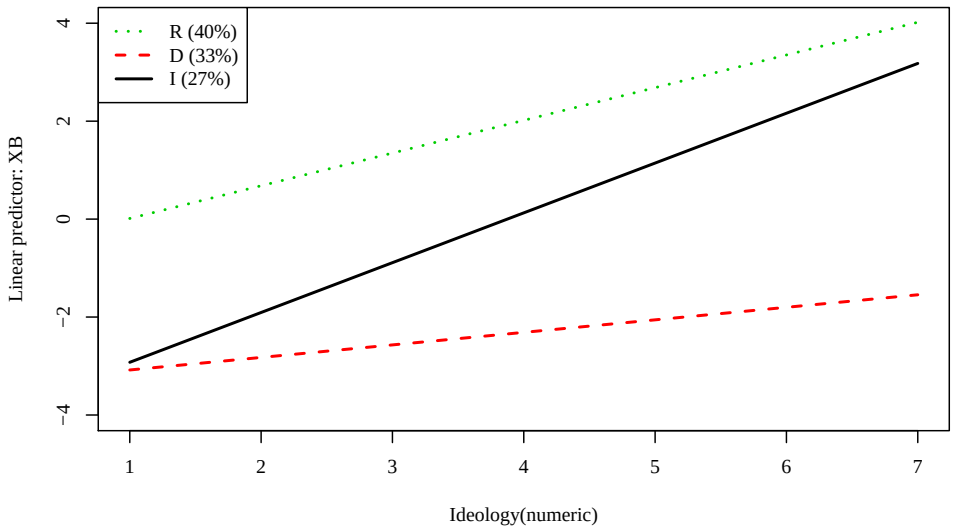
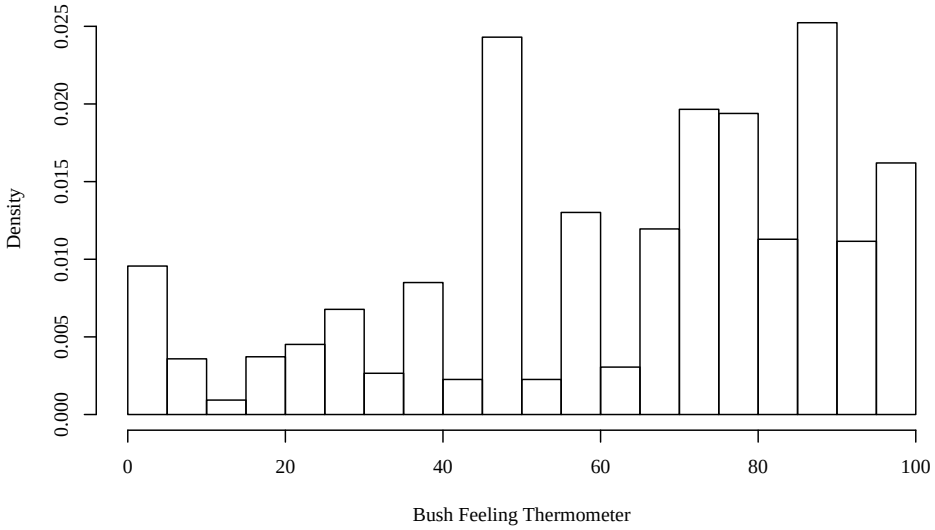


Figure 9: Histogram of the Bush feeling thermometer

```
hist(nes2002$V023010, prob=T, xlab = "Bush Feeling Thermometer",
   breaks = seq(0, 100, by = 5), main="")
```




```
Call:
glm(formula = bushvote ~ V023010, family = binomial, data = nes2002)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.4632  -0.6273   0.3141   0.5978   3.4230

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -6.035992   0.416477  -14.49   <2e-16 ***
V023010      0.090204   0.005713   15.79   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1284.91  on 930  degrees of freedom
Residual deviance:  745.54  on 929  degrees of freedom
  (580 observations deleted due to missingness)
AIC: 749.54

Number of Fisher Scoring iterations: 5
```

A presentable table for these models is provided in Table 4, which also includes the probit model for comparison.

```
glmthermo1 <- glm(bushvote ~ V023010, data=nes2002,
   family=binomial)
summary(glmthermo1)
```

Table 4: Bush Thermometer Models

```
outreg(list("OLS" = olsthermo1, "Logistic" = glmthermo1, "Probit"
   = glmthermo2), tight=FALSE, varLab = vl)
```

	OLS		Logistic		Probit	
	Estimate	(S.E.)	Estimate	(S.E.)	Estimate	(S.E.)
(Intercept)	-0.303***	(0.033)	-6.036***	(0.416)	-3.252***	(0.207)
Bush Therm	0.013***	(0.000)	0.090***	(0.006)	0.049***	(0.003)
N	931		931		931	
RMSE	0.369					
R^2	0.453					
Deviance			745.545		757.607	
$-2LLR(Model\chi^2)$			539.365***		527.303***	

$*p \leq 0.05** p \leq 0.01***p \leq 0.001$

```
Call:
glm(formula = bushvote ~ V023010, family = binomial, data = nes2002)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.4632  -0.6273   0.3141   0.5978   3.4230

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -6.035992   0.416477  -14.49   <2e-16 ***
V023010      0.090204   0.005713   15.79   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1284.91  on 930  degrees of freedom
Residual deviance:  745.54  on 929  degrees of freedom
  (580 observations deleted due to missingness)
AIC: 749.54

Number of Fisher Scoring iterations: 5
```

Figure 10 illustrates how, in 2003, I plotted the relationship between the respondent's "thermometer scale score" for Pres Bush and the vote. The OLS and Logistic regression lines are superimposed on the plot.

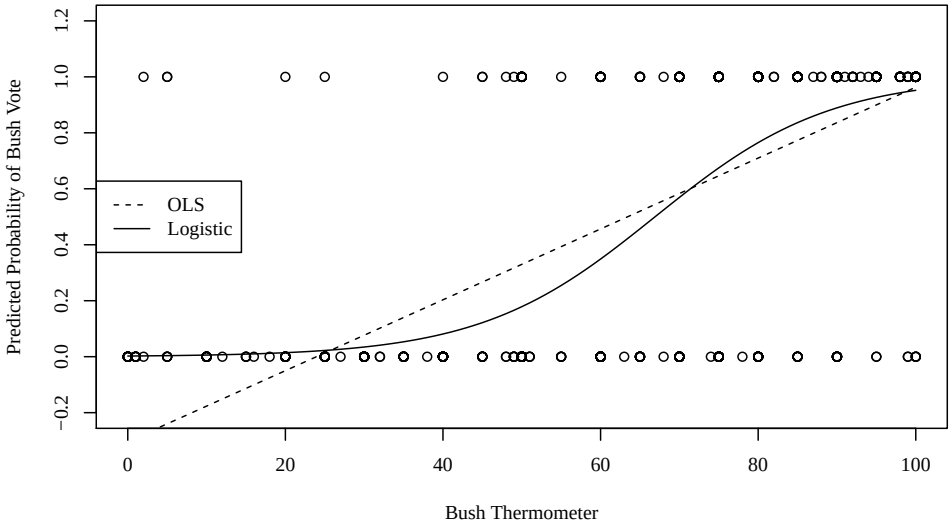
5 The "Full Model"??

We have more predictors available. We might as well use them. I'll leave party as a categorical predictor, but for the moment we leave the rest as if they were reasonably treated as numeric. The full model includes the Bush thermometer variable (V0234010), ideology (V023027), the 2 party variables, education (V023131), and the state of the economy (V023027).

```
bushlogit5 <- glm(bushvote ~ V023010 + ideoln + partyid + useconn
   + educn , data=nes2002, family=binomial(link=logit), model=TRUE)
summary(bushlogit5)
```

Figure 10: Thermometer model: Logistic and OLS Predicted Probabilities

```
nd <- data.frame(V023010 =seq(0,100))
nd$glmpred1 <- predict(glmthermo1, newdata=nd, type="response")
nd$olspred1 <- predict(olsthermo1, newdata=nd)
plot(nes2002$V023010, nes2002$bushvote, ylim=c(-0.2,1.2),
   xlim=c(0,100), xlab="Bush Thermometer", ylab="Predicted
   Probability of Bush Vote")
lines(glmpred1 ~ V023010, data = nd, lty = 1)
lines(olspred1 ~ V023010, data = nd, lty = 2)
legend("left", legend = c("OLS", "Logistic"), lty = c(2,1))
```



```
Call:
glm(formula = bushvote ~ V023010 + ideoln + partyid + useconn +
    educn, family = binomial(link = logit), data = nes2002, model = TRUE)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.7926  -0.2907   0.1455   0.3545   3.3124

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -5.01178    1.01087  -4.958 7.13e-07 ***
V023010      0.06636    0.00784   8.464  < 2e-16 ***
ideoln       0.37047    0.10944   3.385 0.000712 ***
partyidD    -2.53371    0.32075  -7.899 2.80e-15 ***
partyidR     1.29471    0.33167   3.904 9.48e-05 ***
useconn     -0.05464    0.12296  -0.444 0.656800
educn       -0.07259    0.08444  -0.860 0.389932
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1066.72  on 774  degrees of freedom
Residual deviance:  404.64  on 768  degrees of freedom
  (736 observations deleted due to missingness)
AIC: 418.64

Number of Fisher Scoring iterations: 6
```

Outreg summarizes that as:

```
outreg(list("Full Model" = bushlogit5), tight=F, varLab = vl)
```

	Full Model	
	Estimate	(S.E.)
(Intercept)	-5.012***	(1.011)
Bush Therm	0.066***	(0.008)
Ideology	0.370***	(0.109)
Party Dem.	-2.534***	(0.321)
Party Rep.	1.295***	(0.332)
US economy	-0.055	(0.123)
Education	-0.073	(0.084)
N	775	
Deviance	404.643	
$-2LLR(Model\chi^2)$	662.072***	

$*p \leq 0.05** p \leq 0.01***p \leq 0.001$

It is often useful to place the full and partial models side by side. An example would be

```
outreg(list("Full Model"=bushlogit5, "Ideology Only"=bushideoglm1,
 "Ideol. Party Dummies" = bushideoglm2, "Ideol. w/party id" =
    bushideoglm3, "Interaction" = bushideoglm4),
 tight=T, varLab = vl)
```

	Full Model Estimate (S.E.)	Ideology Only Estimate (S.E.)	Ideol. Party Dummies Estimate (S.E.)	Ideol. w/party id Estimate (S.E.)	Interaction Estimate (S.E.)
(Intercept)	-5.012***	-3.952***	-2.382***	-2.529***	-3.941***
	(1.011)	(0.317)	(0.378)	(0.395)	(0.677)
Bush Therm	0.066***	.	.	.	.
	(0.008)				
Ideology	0.370***	0.952***	0.656***	0.670***	1.017***
	(0.109)	(0.070)	(0.087)	(0.090)	(0.162)
Party Dem.	-2.534***	.	.	-2.575***	0.605
	(0.321)			(0.281)	(0.936)
Party Rep.	1.295***	.	.	1.866***	3.287**
	(0.332)			(0.291)	(1.123)
US economy	-0.055	.	.	.	.
	(0.123)				
Education	-0.073	.	.	.	.
	(0.084)				
Repub.	.	.	1.782***	.	.
			(0.287)		
Democ.	.	.	-2.661***	.	.
			(0.276)		
ideoln:partyidD	.	.	.	.	-0.761***
					(0.223)
ideoln:partyidR	.	.	.	.	-0.350
					(0.246)
N	775	809	809	781	781
Deviance	404.643	833.913	553.476	519.677	507.238
$-2LLR(Model\chi^2)$	662.072***	276.885***	557.322***	555.014***	567.454***

$*p \leq 0.05** p \leq 0.01***p \leq 0.001$

Note: I changed to the tight format in order to save horizontal space.

Nested model test

The parameter model=TRUE requests glm to keep a copy of the data set that was used. Because we keep a copy of the data set, it is easy to do an ANOVA style nested model test.

Suppose we wonder if it is "save" to remove the dummies for party membership and the US economy variable from the full model. (I don't mean we should do that, but somebody might wonder *what if.*)

Fit the "smaller" model that restricts the omitted variables (useecon and partyid) by assuming their coefficients are equal to 0.

```
bushlogit5b <- glm(bushvote ~ V023010 + ideoln + educn,
 data=model.frame(bushlogit5), family=binomial(link=logit),
    model=TRUE)
```

The likelihood ratio test is performed by the R anova method for generalized linear models.

```
anova(bushlogit5, bushlogit5b, test = "Chisq")
```

```
Analysis of Deviance Table

Model 1: bushvote ~ V023010 + ideoln + partyid + useconn + educn
Model 2: bushvote ~ V023010 + ideoln + educn
  Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
1       768     404.64
2       771     556.54 -3   -151.9 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The result says the 2 models are statistically significantly different, so it is not save to remove both of those variables.

6 What about using ordinal variables as numeric predictors?[2]

This is an age-old controversy. Is a 7 point Likert scale actually a numeric variable, or is it just a categorical ordering?

My strategy here is to avoid the general philosophical debate and convert the debate to more practical terms. My answer is this: test whether the categorical values are meaningfully different in their impact on the prediction, and then find out if the impact of the categories is actually "linear". If it is, then treat the predictor as a number. Otherwise, treat it as categories.

6.1 Education

Why bother? Well, the full model seems to say that education's coefficient in the Bush vs Gore decision is not meaningfully different from 0. That may be a mistake, however, because we have used the 7 category data as if it were a meaningful numeric scale. Perhaps instead of a linear 1-2-3 effect, the impact of adjusting education across the 7 categories is something like 0.5, 0.8, 0.8, 1.0, 1.0, 1.0? If the effect of education happens in discrete lumps, then we may need to combine the people who finished some college with the people who finished with a 2 year Associate's degree, for example. Or perhaps we should combine college graduates and post-graduate degree holders? (Joke: How much smarter do you need to get before you see Al Gore was right?).

The method here is as follows. Wit the same "big model" but replace the numeric education variable with the unordered categorical variable, `educf`.

ANOVA test to find out if any of the education levels matter

The full model is

$$Pr(y_i = 1) = \frac{1}{1 + e^{-\eta_i}} \tag{3}$$

[2] This is the Ian Ostrander memorial section

where

$$\eta_i = \beta_0 + \beta_1 BushTherm + \beta_2 Ideology + \beta_3 Democ. + \beta_4 Repub. + \beta_5 USEcon + \tag{4}$$

$$\beta_6 educ2_i + \beta_7 educ3_i + \beta_8 educ4_i + \beta_9 educ5_i + \beta_{10} educ6_i + \beta_{11} educ7_i \tag{5}$$

When we estimate education as unordered factor variable, we are introducing a series of "dummy variables". The lowest level of education, the elementary-school cases, are in the intercept, and the following are estimated as contrasts against that low level. We can get estimates for each of the 6 different levels (could suppress intercept to see explicit lowest category).

```
bushlogit6 <- glm(bushvote ~ V023010 + ideoln + partyid + useconn
   + educf, data=nes2002, family=binomial(link=logit), model=TRUE)
summary(bushlogit6)
```

```
Call:
glm(formula = bushvote ~ V023010 + ideoln + partyid + useconn +
    educf, family = binomial(link = logit), data = nes2002, model = TRUE)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.7735  -0.2855   0.1374   0.3442   3.3110

Coefficients:
                 Estimate Std. Error z value Pr(>|z|)
(Intercept)     -7.059758   1.399588  -5.044 4.55e-07 ***
V023010          0.066918   0.007931   8.437  < 2e-16 ***
ideoln           0.386006   0.111888   3.450 0.000561 ***
partyidD        -2.629314   0.333802  -7.877 3.36e-15 ***
partyidR         1.307908   0.338310   3.866 0.000111 ***
useconn         -0.088049   0.127602  -0.690 0.490176
educfSome HS     1.432101   1.344566   1.065 0.286829
educfHS          2.001368   1.106675   1.808 0.070536 .
educfSome coll.  2.451995   1.118111   2.193 0.028309 *
educfAA          1.420641   1.142753   1.243 0.213803
educfBA          1.287433   1.099674   1.171 0.241703
educfGrad        1.887883   1.118700   1.688 0.091494 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1066.72  on 774  degrees of freedom
Residual deviance:  392.39  on 763  degrees of freedom
  (736 observations deleted due to missingness)
AIC: 416.39

Number of Fisher Scoring iterations: 6
```

What does this show? Should we conclude that education plays a role, or not? A couple of categories "seem to matter", but what is the bottom line?

ANOVA test deletion of variables in a nested model

If we state the null hypothesis that "all education coefficients are 0" and let the alternative be "at least one of the education coefficients is not 0," then we should conduct a Likelihood Ratio test comparing the two models. The built-in function `drop1` will do all of the work:

```
drop1(bushlogit6, test="Chisq")
```

```
Single term deletions

Model:
bushvote ~ V023010 + ideoln + partyid + useconn + educf
        Df Deviance    AIC     LRT  Pr(>Chi)
<none>       392.39 416.39
V023010  1   494.32 516.32 101.927 < 2.2e-16 ***
ideoln   1   404.42 426.42  12.027 0.0005243 ***
partyid  2   546.73 566.73 154.338 < 2.2e-16 ***
useconn  1   392.87 414.87   0.479 0.4889513
educf    6   405.39 417.39  12.992 0.0431597 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Note that the 'drop1' groups together all of the levels of the factor variable and it addresses the question "does this variable make a difference *at all*." In other words, it is conducting the Likelihood Ratio test that you really want if you are deciding whether education plays a role in predicting voting. This test says that at least some of the categories are different from 0 in their effect.

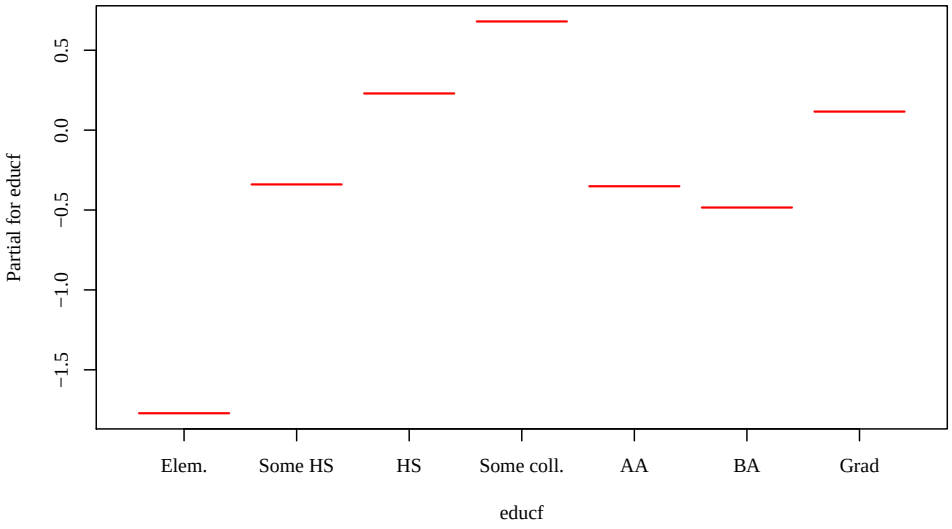
The drop1 is identical to the test we would conduct by hand, if we fitted a smaller model by omitting one predictor at a time and running anova() . The anova function performs the likelihood ratio test because we specified test="Chisq").

Is the education effect linear?

If the effect of education were linear, then the plot of the predicted values would appear as evenly spaced steps. That does not seem to be the case in Figure 11.

Figure 11: Termplot display of Education "step" effects

```
termplot(bushlogit6, terms="educf")
```



The anova test on the categorical predictor "educf" indicates that "education matters." Now we want to know if the effect is a linear step sequence of even steps. The graph makes me expect that the linear model is wrong. One quick hint is to compare the "residual deviance" values of the two models. What do you see?

We can formally test that conclusion, however. Conduct a likelihood ratio test, comparing the "full" model that uses the factor coding of education with the "reduced" model which assumes that the different levels have the same effects. Note how ironic it is that the restricted model in this comparison is the one I called the full model in a previous section.

```
anova(bushlogit5, bushlogit6, test="Chisq")
```

```
Analysis of Deviance Table

Model 1: bushvote ~ V023010 + ideoln + partyid + useconn + educn
Model 2: bushvote ~ V023010 + ideoln + partyid + useconn + educf
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1       768     404.64
2       763     392.39  5    12.25  0.03152 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Followup: compressing categories

In the Figure 11, we might have evidence for the idea that categories 5, 6, and 7 have the same effect. We can construct a new predictor that puts those together. In rockchalk I put in a special purpose function for this called combineLevels, but the general purpose plyr::mapvalues function would also do the job. (If you say, "This looks like a fool's errand," I agree, but lets work out the example.)

```
nes2002$educf2 <- combineLevels(nes2002$educf, levs=c(5,6,7),
   "Higher Ed")
```

```
The original levels Elem. Some HS HS Some coll. AA BA Grad
have been replaced by Elem. Some HS HS Some coll. Higher Ed
```

```
table("educ: reduced" = nes2002$educf2, "educ: 7 categories" =
   nes2002$educf, exclude = NULL)
```

```
             educ: 7 categories
educ: reduced Elem. Some HS  HS Some coll.  AA  BA Grad <NA>
   Elem.          36       0   0          0   0   0    0    0
   Some HS         0      70   0          0   0   0    0    0
   HS              0       0 399          0   0   0    0    0
   Some coll.      0       0   0        313   0   0    0    0
   Higher Ed       0       0   0          0 155 347  178    0
   <NA>            0       0   0          0   0   0    0   13
```

```
bushlogit6c <- glm(bushvote ~ V023010 + ideoln + partyid + educf2
   + useconn, data=nes2002, family=binomial(link=logit))
summary(bushlogit6c)
```

```
Call:
glm(formula = bushvote ~ V023010 + ideoln + partyid + educf2 +
    useconn, family = binomial(link = logit), data = nes2002)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.7599  -0.2879   0.1383   0.3350   3.2901

Coefficients:
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)      -6.952875   1.387320  -5.012 5.39e-07 ***
V023010           0.065777   0.007869   8.359  < 2e-16 ***
ideoln            0.390721   0.111757   3.496 0.000472 ***
partyidD         -2.662545   0.332474  -8.008 1.16e-15 ***
partyidR          1.257030   0.335806   3.743 0.000182 ***
educf2Some HS     1.440676   1.337084   1.077 0.281268
educf2HS          2.007157   1.098655   1.827 0.067711 .
educf2Some coll.  2.457434   1.110353   2.213 0.026884 *
educf2Higher Ed   1.501258   1.076607   1.394 0.163186
useconn          -0.094573   0.127159  -0.744 0.457038
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1066.72  on 774  degrees of freedom
Residual deviance:  394.69  on 765  degrees of freedom
  (736 observations deleted due to missingness)
AIC: 414.69

Number of Fisher Scoring iterations: 6
```

The null hypothesis is that we prefer to keep the restriction so that the last 3 levels of education have the same effect, while the alternative is that we need to keep the larger model which estimates separate coefficients. We then test that simplified categorical model against the previous fit that included the original factor variable.

```
anova(bushlogit6c, bushlogit6, test="Chisq")
```

```
Analysis of Deviance Table

Model 1: bushvote ~ V023010 + ideoln + partyid + educf2 + useconn
Model 2: bushvote ~ V023010 + ideoln + partyid + useconn + educf
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1       765     394.69
2       763     392.39  2   2.2979    0.317
```

Well, the conclusion is that the model that uses the simpler set of categories for the education variable is not noticeably worse than the one that estimates separate coefficient for each category.

Where does this leave us with the education variable, however. It appears we might want to squish more levels together. Or perhaps we should look up and realize that education, no matter how we code it, is not very tightly linked to how people voted in 2000.

Enjoy this drop1 output, which seems to say that this 4-level education variable has coefficients that differ from 0.

```
drop1(bushlogit6c, test="Chisq")
```

```
Single term deletions
```

```
Model:
bushvote ~ V023010 + ideoln + partyid + educf2 + useconn
        Df Deviance    AIC     LRT  Pr(>Chi)
<none>       394.69 414.69
V023010  1   494.46 512.46  99.773 < 2.2e-16 ***
ideoln   1   407.09 425.09  12.399 0.0004295 ***
partyid  2   550.35 566.35 155.656 < 2.2e-16 ***
educf2   4   405.39 417.39  10.694 0.0302231 *
useconn  1   395.25 413.25   0.556 0.4557503
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

6.2 Ideology

I found myself wondering if the ideology variable should also be given the "factor" consideration. Planning for this phase, I created 2 versions of the categorical variable. One is an ordinary unordered factor, `ideof`.

The other variable, `ideolo`, is an ordered factor. When R regression routines find a predictor that is an ordered factor, they change the contrast codes to use orthogonal polynomials.

Treatment contrasts

The full model re-fit with the categorical unordered variable, `ideolf`, provides "dummy" variable estimates for the highest 6 levels of ideology.

```
bushideolrm7f <- glm(bushvote ~ V023010 + ideolf + partyid + educn
   + useconn, data=nes2002, family=binomial(link=logit))
summary(bushideolrm7f)
```

```
Call:
glm(formula = bushvote ~ V023010 + ideolf + partyid + educn +
    useconn, family = binomial(link = logit), data = nes2002)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.9006  -0.2739   0.1457   0.3387   3.1918

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -2.973156   1.408094  -2.111  0.03473 *
V023010      0.065749   0.007878   8.346  < 2e-16 ***
ideolfL     -1.467586   1.198261  -1.225  0.22066
ideolfSL    -0.823231   1.164518  -0.707  0.47961
ideolfM     -0.763800   1.117647  -0.683  0.49435
ideolfSC    -0.277315   1.145282  -0.242  0.80867
ideolfC      0.505559   1.154340   0.438  0.66141
ideolfEC    -0.014066   1.344559  -0.010  0.99165
partyidD    -2.553675   0.327330  -7.802 6.12e-15 ***
partyidR     1.241612   0.337076   3.683  0.00023 ***
educn       -0.062552   0.086518  -0.723  0.46969
useconn     -0.041500   0.125335  -0.331  0.74056
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1066.72  on 774  degrees of freedom
Residual deviance:  399.85  on 763  degrees of freedom
```

```
  (736 observations deleted due to missingness)
AIC: 423.85

Number of Fisher Scoring iterations: 6
```

A formal test that compares the "factor coding" against the "linear coding" is as follows:

```
anova(bushlogit5, bushideolrm7f, test="Chisq")
```

```
Analysis of Deviance Table

Model 1: bushvote ~ V023010 + ideoln + partyid + useconn + educn
Model 2: bushvote ~ V023010 + ideolf + partyid + educn + useconn
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1       768     404.64
2       763     399.85  5   4.7916   0.4418
```

The result difference in the predictive power of the two models is not statistically significant, so we probably ought to go ahead and use the "linear coding".

Would you stop with that conclusion? I think you probably should. You'd wonder why the lowest level of ideology appears to have such a strange spike. Look back at the descriptive table for the ideology variable. Notice that there are 12 respondents, only 12, who have the lowest score, which is 1. The anova test indicates that the aberrant estimate for that group is not "statistically out of line," at least not far enough out of line to make us think we need to worry about treating ideology as a factor.

orthopoly contrasts

The fit of the model that uses the ordinal variable, `ideolo`, is superficially different. Instead of fitting the intercept as a baseline group and then adding 6 dummy variables, the orthogonal polynomial contrasts create predictor columns that are, by definition, uncorrelated with each other (hence, "orthogonal"). I made some notes a while ago about orthogonal polynomial contrasts (look in http://pj.freefaculty.org/R/WorkingExamples for a file named "regression-orderedFactors...").

These contrasts appear to be grossly different in interpretation, but the difference is only skin deep. The ordinal model triggers the usage of "orthopoly" contrasts, an alternative way of creating the comparisons that are used.

```
bushideolrm7o <- glm(bushvote ~ V023010 + ideolo + partyid + educn
   + useconn, data=nes2002, family=binomial(link=logit))
summary(bushideolrm7o)
```

```
Call:
glm(formula = bushvote ~ V023010 + ideolo + partyid + educn +
    useconn, family = binomial(link = logit), data = nes2002)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.9006  -0.2739   0.1457   0.3387   3.1918

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -3.378932   0.981070  -3.444 0.000573 ***
V023010      0.065749   0.007878   8.346  < 2e-16 ***
ideolo.L     0.840973   0.805728   1.044 0.296604
```

```
ideolo.Q     0.685914   0.751173   0.913 0.361177
ideolo.C    -1.034145   0.625084  -1.654 0.098044 .
ideolo^4     0.081279   0.471657   0.172 0.863181
ideolo^5    -0.564863   0.393908  -1.434 0.151573
ideolo^6     0.148891   0.301841   0.493 0.621817
partyidD    -2.553675   0.327330  -7.802 6.12e-15 ***
partyidR     1.241612   0.337076   3.683 0.000230 ***
educn       -0.062552   0.086518  -0.723 0.469687
useconn     -0.041500   0.125335  -0.331 0.740558
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1066.72  on 774  degrees of freedom
Residual deviance:  399.85  on 763  degrees of freedom
  (736 observations deleted due to missingness)
AIC: 423.85

Number of Fisher Scoring iterations: 6
```

It *looks* different, but if you look at the models side-by-side, there are hints that they are actually interchangeable:

```
outreg(list("ideolf"=bushideolrm7f, "ideolo"=bushideolrm7o), tight
   = F, varLab = vl)
```

	ideolf		ideolo	
	Estimate	(S.E.)	Estimate	(S.E.)
(Intercept)	-2.973*	(1.408)	-3.379***	(0.981)
Bush Therm	0.066***	(0.008)	0.066***	(0.008)
ideolfL	-1.468	(1.198)	.	
ideolfSL	-0.823	(1.165)	.	
ideolfM	-0.764	(1.118)	.	
ideolfSC	-0.277	(1.145)	.	
ideolfC	0.506	(1.154)	.	
ideolfEC	-0.014	(1.345)	.	
Party Dem.	-2.554***	(0.327)	-2.554***	(0.327)
Party Rep.	1.242***	(0.337)	1.242***	(0.337)
Education	-0.063	(0.087)	-0.063	(0.087)
US economy	-0.042	(0.125)	-0.042	(0.125)
ideolo.L	.		0.841	(0.806)
ideolo.Q	.		0.686	(0.751)
ideolo.C	.		-1.034	(0.625)
ideolo.4	.		0.081	(0.472)
ideolo.5	.		-0.565	(0.394)
ideolo.6	.		0.149	(0.302)
N	775		775	
Deviance	399.852		399.852	
$-2LLR(Model\chi^2)$	666.863***		666.863***	

$*p \leq 0.05** p \leq 0.01***p \leq 0.001$

Note the likelihood ratio test with the ordinal version of the factor leads to the exact same conclusion that we reached when we tested with the previous model:

```
anova(bushlogit5, bushideolrm7o, test="Chisq")
```

```
Analysis of Deviance Table

Model 1: bushvote ~ V023010 + ideoln + partyid + useconn + educn
Model 2: bushvote ~ V023010 + ideolo + partyid + educn + useconn
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1       768     404.64
2       763     399.85  5   4.7916   0.4418
```

The hypothesis test is the same, whether we use `ideolo` or `ideolf`.

You might wonder, "what is an orthogonal polynomial". Here is a hint:

```
x <- 1:7
xf <- factor(x)
contrasts(xf)
```

```
  2 3 4 5 6 7
1 0 0 0 0 0 0
2 1 0 0 0 0 0
3 0 1 0 0 0 0
4 0 0 1 0 0 0
5 0 0 0 1 0 0
6 0 0 0 0 1 0
7 0 0 0 0 0 1
```

When the regression routine finds a variable coded like xf, it creates 6 columns of "dummy variables" as we see here. If an observed score on the factor is 1, all of the dummies are 0's, meaning the case is "in the intercept." If x is 2 then the values of the dummy variables are pulled from row 2, $(1, 0, 0, 0, 0, 0)$. If x happens to be a 5, then $(0, 0, 0, 1, 0, 0)$ is the dummy variable vector. Hopefully, that's what you expect.

In comparison, these are the predictor rows that are substituted for the observed values of the ordered factor:

```
xo <- factor(x, ordered = TRUE)
contrasts(xo)
```

```
                .L            .Q            .C          ^4           ^5          ^6
[1,] -5.669467e-01  5.455447e-01 -4.082483e-01  0.2417469 -1.091089e-01  0.03289758
[2,] -3.779645e-01  9.690821e-17  4.082483e-01 -0.5640761  4.364358e-01 -0.19738551
[3,] -1.889822e-01 -3.273268e-01  4.082483e-01  0.0805823 -5.455447e-01  0.49346377
[4,]  2.098124e-17 -4.364358e-01  3.021644e-17  0.4834938 -9.751389e-16 -0.65795169
[5,]  1.889822e-01 -3.273268e-01 -4.082483e-01  0.0805823  5.455447e-01  0.49346377
[6,]  3.779645e-01  0.000000e+00 -4.082483e-01 -0.5640761 -4.364358e-01 -0.19738551
[7,]  5.669467e-01  5.455447e-01  4.082483e-01  0.2417469  1.091089e-01  0.03289758
```

These are quite different in appearance, but they may help diagnose nonlinear trends in some models. Unfortunately, in this case, they all say "we can't detect anything."

Note the interesting quirk about ideology

In the model that uses ideology as a numeric predictor, we find the coefficient is statistically significantly different from 0. However, when we take it as a categorical variable, then we are not

able to say with much confidence that any two categories are different from each other. That might make us think we should drop the ideology variable, but the likelihood ratio test (drop1) indicates that ideology coefficients are probably not all equal to 0.

```
drop1(bushideolrm7f, test = "Chisq")
```

```
Single term deletions

Model:
bushvote ~ V023010 + ideolf + partyid + educn + useconn
        Df Deviance    AIC     LRT Pr(>Chi)
<none>       399.85 423.85
V023010  1   499.05 521.05  99.196  < 2e-16 ***
ideolf   6   416.28 428.28  16.428  0.01163 *
partyid  2   547.34 567.34 147.487  < 2e-16 ***
educn    1   400.38 422.38   0.524  0.46898
useconn  1   399.96 421.96   0.110  0.74024
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

That, in a nutshell, is the justification for the nested model hypothesis test procedure. None of the separate categories of ideology seem to matter, but the null hypothesis that all of the ideology of parameters are all equal to 0 is rejected. This result is somewhat unusual in my experience. Usually, at least one of the dummy variables is statistically significant.

6.3 Can we save the "US economy" variable?

The answer will be no, it seems.

The numeric coding says there is not a detectable linear relationship for the US economy variable, `useconn`.

I hate to admit the economy does not matter. Maybe the effect is not linear! Lets take the "full" model and replace the numeric US economy variable by a categorical `usecono`:

```
bushlogit8 <- glm(bushvote ~ V023010 + ideoln + partyid + usecono
   + educn, data=nes2002, family=binomial(link=logit), model=TRUE)
summary(bushlogit8)
```

```
Call:
glm(formula = bushvote ~ V023010 + ideoln + partyid + usecono +
    educn, family = binomial(link = logit), data = nes2002, model = TRUE)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.7987  -0.2923   0.1443   0.3538   3.3119

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -5.179402   0.839957  -6.166 6.99e-10 ***
V023010      0.066318   0.007851   8.447  < 2e-16 ***
ideoln       0.370226   0.109493   3.381 0.000721 ***
partyidD    -2.532259   0.321067  -7.887 3.10e-15 ***
partyidR     1.294283   0.331767   3.901 9.57e-05 ***
usecono.L   -0.124297   0.457457  -0.272 0.785842
usecono.Q   -0.035813   0.353533  -0.101 0.919312
educn       -0.072949   0.084501  -0.863 0.387980
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1066.72  on 774  degrees of freedom
Residual deviance:  404.63  on 767  degrees of freedom
  (736 observations deleted due to missingness)
AIC: 420.63

Number of Fisher Scoring iterations: 6
```

I think we found out in the previous section that we can't read the orthogonal contrast estimates from the table. We have to rely on formal hypothesis tests (or, possibly, give in to temptation and create an ordinary factor variable useconf that will use the treatment contrasts we have come to enjoy so much).

The `drop1` function will check all of the variables, but we are really only interested in `usecono`, of course

```
drop1(bushlogit8, test = "Chisq")
```

```
Single term deletions

Model:
bushvote ~ V023010 + ideoln + partyid + usecono + educn
         Df Deviance    AIC     LRT  Pr(>Chi)
<none>        404.63 420.63
V023010   1   505.94 519.94 101.306 < 2.2e-16 ***
ideoln    1   416.24 430.24  11.606 0.0006573 ***
partyid   2   555.01 567.01 150.373 < 2.2e-16 ***
usecono   2   404.84 416.84   0.208 0.9010638
educn     1   405.38 419.38   0.748 0.3870909
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

I suppose we could check to make sure that test is doing the "right thing". To check that estimate, we need to fit the same model, but omit usecono, and then compare with a likelihood-ratio test.

```
bushlogit8r <- glm(bushvote ~ V023010 + ideoln + partyid + educn,
   data=model.frame(bushlogit8), family=binomial(link=logit))
anova(bushlogit8r, bushlogit8, test = "Chisq")
```

```
Analysis of Deviance Table

Model 1: bushvote ~ V023010 + ideoln + partyid + educn
Model 2: bushvote ~ V023010 + ideoln + partyid + usecono + educn
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1       769     404.84
2       767     404.63  2  0.20836   0.9011
```

Fortunately for us, the result is equivalent to the value reported in `drop1`. However, it is still a little disappointing that the economy does not matter.

Finally, it might be that the model fitted with the categorical version is somehow better than the linear coding used in the "full" model. But alas, it is not so.

```
anova(bushlogit5, bushlogit8, test = "Chisq")
```

```
Analysis of Deviance Table

Model 1: bushvote ~ V023010 + ideoln + partyid + useconn + educn
Model 2: bushvote ~ V023010 + ideoln + partyid + usecono + educn
```

```
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1       768     404.64
2       767     404.63  1  0.01025   0.9194
```

7 Compare Logit and Probit links

7.1 Estimating a Probit model: glm with binomial(probit)

With R's glm, the distinction between logit and probit is found in the "link" function, the transformation that is applied to the left hand side to go from "probability" into the scale of the linear predictor. The Logistic model was estimated with the command

```
bushideoglm1 <- glm(bushvote~ideoln, family=binomial, data=nes2002)
```

To fit a probit model, one in which the random error is normal, we make only one small change in the command.

```
bushideoProbit1 <- glm(bushvote~ideoln,
   family=binomial(link=probit), data=nes2002)
summary(bushideoProbit1)
```

```
Call:
glm(formula = bushvote ~ ideoln, family = binomial(link = probit),
    data = nes2002)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.4018  -0.7808   0.5713   0.8723   2.5347

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -2.30389    0.17306  -13.31   <2e-16 ***
ideoln       0.55630    0.03768   14.76   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1110.80  on 808  degrees of freedom
Residual deviance:  836.02  on 807  degrees of freedom
  (702 observations deleted due to missingness)
AIC: 840.02

Number of Fisher Scoring iterations: 4
```

7.2 OLS, Logit and Probit Side by Side

Ideology fits side by side.

```
outreg(list(ols1, bushideoglm1, bushideoProbit1), tight=T,
   showAIC=T, modelLabels=c("OLS","Logit","Probit"), varLab = vl)
```

	OLS Estimate (S.E.)	Logit Estimate (S.E.)	Probit Estimate (S.E.)
(Intercept)	-0.248***	-3.952***	-2.304***
	(0.045)	(0.317)	(0.173)
Ideology	0.181***	0.952***	0.556***
	(0.010)	(0.070)	(0.038)
N	809	809	809
RMSE	0.415		
R^2	0.304		
Deviance		833.913	836.019
$-2LLR(Model\chi^2)$		276.885***	274.778***
AIC	876.564	837.913	840.019

$*p \leq 0.05** p \leq 0.01***p \leq 0.001$

The Full Model fitted with all three methods

Compare the OLS, Logit, and Probit fits in the full model.

```
bushprobit5 <- glm(bushvote ~ V023010 + ideoln + partyid + educn +
   useconn, data=nes2002, family=binomial(link=probit), model=T)
summary(bushprobit5)
```

```
Call:
glm(formula = bushvote ~ V023010 + ideoln + partyid + educn +
    useconn, family = binomial(link = probit), data = nes2002,
    model = T)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.8473  -0.3099   0.1113   0.3659   3.5075

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -2.534823   0.531153  -4.772 1.82e-06 ***
V023010      0.035123   0.004026   8.725  < 2e-16 ***
ideoln       0.204836   0.058127   3.524 0.000425 ***
partyidD    -1.420566   0.170647  -8.325  < 2e-16 ***
partyidR     0.698687   0.174842   3.996 6.44e-05 ***
educn       -0.067508   0.045380  -1.488 0.136845
useconn     -0.020976   0.065527  -0.320 0.748882
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1066.72  on 774  degrees of freedom
Residual deviance:  408.67  on 768  degrees of freedom
  (736 observations deleted due to missingness)
AIC: 422.67

Number of Fisher Scoring iterations: 6
```

Outreg summarizes that as:

```
outreg(list(busholS5, bushlogit5, bushprobit5), tight=T,
   showAIC=T, modelLabels=c("OLS","Logit","Probit"), varLab = vl)
```

	OLS Estimate (S.E.)	Logit Estimate (S.E.)	Probit Estimate (S.E.)
(Intercept)	0.060	-5.012***	-2.535***
	(0.081)	(1.011)	(0.531)
Bush Therm	0.006***	0.066***	0.035***
	(0.001)	(0.008)	(0.004)
Ideology	0.035***	0.370***	0.205***
	(0.009)	(0.109)	(0.058)
Party Dem.	-0.357***	-2.534***	-1.421***
	(0.029)	(0.321)	(0.171)
Party Rep.	0.201***	1.295***	0.699***
	(0.030)	(0.332)	(0.175)
Education	-0.007	-0.073	-0.068
	(0.007)	(0.084)	(0.045)
US economy	-0.007	-0.055	-0.021
	(0.010)	(0.123)	(0.066)
N	775	775	775
RMSE	0.298		
R^2	0.644		
adj R^2	0.641		
Deviance		404.643	408.672
$-2LLR(Model\chi^2)$		662.072***	658.043***
AIC	333.271	418.643	422.672

$*p \leq 0.05** p \leq 0.01***p \leq 0.001$

The ratio of the logit to the probit estimates depends on the assumed value of standard deviation of the error term (as that term arises in the CDF interpretation of the logit and probit models). That standard deviation cannot be estimated, it has to be set by the user to some arbitrary value, and all of the coefficients adjust in response. I extracted the estimated coefficients from the 2 models and then calculated the ratio of the logit to the probit estimates. I expected that the ratios would be almost exactly the same, and that made me suspect I had done something incorrectly. But I don't see the mistake.

```
logitcoef <- coef(bushlogit5)
probitcoef <- coef(bushprobit5)
cbind(logitcoef, probitcoef, ratio=logitcoef/probitcoef)
```

```
             logitcoef  probitcoef     ratio
(Intercept) -5.01178177 -2.53482316 1.9771722
V023010      0.06636326  0.03512340 1.8894317
ideoln       0.37047047  0.20483552 1.8086241
partyidD    -2.53371136 -1.42056620 1.7835926
partyidR     1.29471298  0.69868715 1.8530654
useconn     -0.05463711 -0.06750850 0.8093367
educn       -0.07259333 -0.02097614 3.4607568
```

Lets compare the ratio of b/se(b) and the p values to see what the differences might be:

```
options(digits=3)
logitsummary <- summary(bushlogit5)
probitsummary <- summary(bushprobit5)
```

```
zcompare <- cbind(logitsummary$coefficients[, c(3,4)],
   probitsummary$coefficients[, c(3,4)])
zcompare <- as.data.frame(zcompare)
colnames(zcompare) <- c("logit-z","logit-p", "probit-z",
   "probit-p")
zcompare$ratioz <- zcompare[ ,1] / zcompare[ ,3]
zcompare
```

```
            logit-z  logit-p probit-z probit-p ratioz
(Intercept)  -4.958 7.13e-07    -4.77 1.82e-06  1.039
V023010       8.464 2.58e-17     8.72 2.66e-18  0.970
ideoln        3.385 7.12e-04     3.52 4.25e-04  0.961
partyidD     -7.899 2.80e-15    -8.32 8.46e-17  0.949
partyidR      3.904 9.48e-05     4.00 6.44e-05  0.977
useconn      -0.444 6.57e-01    -1.49 1.37e-01  0.299
educn        -0.860 3.90e-01    -0.32 7.49e-01  2.686
```

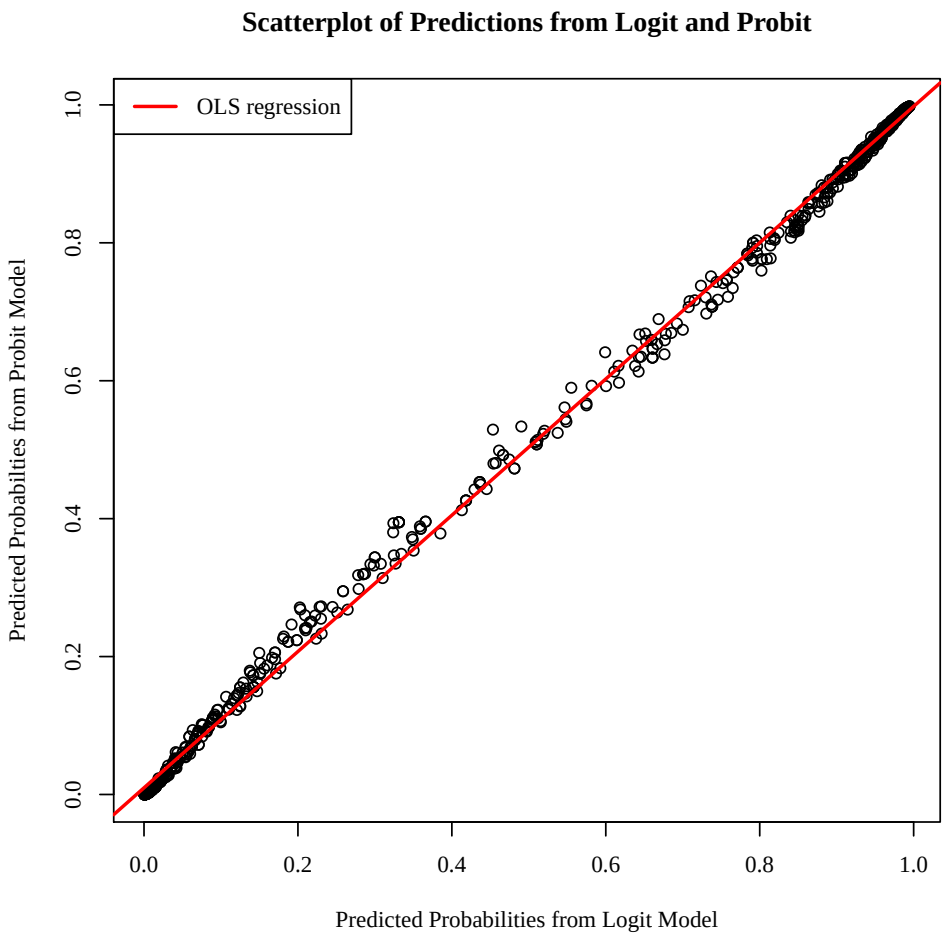
In practice, I have never fitted a model for dichotomous data for which I felt that the choice of the logit or probit link made a substantial difference. The differences here in the estimated ratios of $\hat{\beta}/se(\hat{\beta})$ are not trivial, but they also are not so great as to make us change our conclusions.

Finally, lets get the predicted values from the models and plot them against each other.

```
logitpred <- predict(bushlogit5, type="response")
probitpred <- predict(bushprobit5, type="response")
plot(logitpred, probitpred, xlab="Predicted Probabilities from
   Logit Model", ylab="Predicted Probabilties from Probit Model",
   main="Scatterplot of Predictions from Logit and Probit")
pmod1 <- lm(probitpred ~ logitpred)
summary(pmod1)
```

```
Call:
lm(formula = probitpred ~ logitpred)

Residuals:
     Min       1Q   Median       3Q      Max
-0.04272 -0.01071 -0.00005  0.00619  0.07146

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 0.009694   0.000884      11   <2e-16 ***
logitpred   0.988148   0.001287     768   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.0148 on 773 degrees of freedom
Multiple R-squared:  0.999, Adjusted R-squared:  0.999
F-statistic: 5.9e+05 on 1 and 773 DF,  p-value: <2e-16
```

```
abline(pmod1, col="red", lwd=2)
legend("topleft", legend=c("OLS regression"), col="red", lwd=2)
```



8 ROC

In the notes on logit models, there is a brief section on the ROC curve approach to measuring the quality of the predictions from the model. We want the model to predict 0 when the truth is 0 (True Negative) and we want a prediction of 1 if the observed score is 1 (True Positive).

Take the predicted probabilities from the model, $\hat{p}_i$, and calculate predictions for a cutoff point, τ. Suppose for 3 cases, $\hat{p} = (0.3, 0.6, 0.8)$. In the usual way of thinking about logit models, the prediction dividing point is 0.5 and the model would predict $(0, 1, 1)$. Now we change that story by supposing that the divider τ might be very high, say 0.9, in which the model predicts $(0, 0, 0)$. However, as we dial the divider down, the predictions change, first at $\tau = 0.8$, we predict $(0, 0, 1)$, and the predictions stay the same until the divider hits 0.6,when the prediction changes to $(0, 1, 1)$,and then when the divider falls to 0.3, then the predictions are $(1, 1, 1)$.

I'll write the prediction for a case as a function of the cutoff value:

$$\hat{y}_i(\tau) = \begin{cases} 1 & \hat{p}_i \geq \tau \\ 0 & else \end{cases} \tag{6}$$

For a vector of predictions, we can calculate the percentages predicted correctly, the true positive rate (TPR), the true negative rate (TNR), the false positive rate (FPR) and the false negative rate (FNR).

Generally, we experience a tradeoff between specificity, the ability to correctly predict 0's, and sensitivity, the ability to predict 1's. A model that predicts all 0's has fantastic specificity, but poor sensitivity, while a model that predicts all 1's is sensitive, but not selective. If this were a medical diagnostic test, for example, we might be forced into a situation of using a test that is either too "jumpy" (we tell many people they have cancer, but in fact they do not) or too insensitive (we don't tell people they have cancer when we ought to).

A common graph for this process is the ROC curve. We vary τ from 1 to 0 (predictions begin as all 0's, and end up at all 1's). The ROC curve in Figure 12 explores the nature of possible predictions by plotting the change as τ changes from 1 to 0. Start at the bottom left, where the model predicts 0 for every case. The horizontal value, specificity, is 1.0, meaning that there are no "false positives", and sensitivity is 0, meaning that none of the true positives were predicted correctly. This point at the bottom left happens when we make the model predict 0 for every row in the data. As we move the cutpoint into the range of predicted probabilities, the predictions change for some cases from 0 to 1.

Hopefully, the cases for which we change the prediction to 1 are correct, meaning our TNR is not dropping off too fast, but the TPR is increasing rapidly. A model that fits the data very accurately will have an ROC curve like the one in Figure 12. When the value of τ reaches 0.5, we hope that the curve will be in the top left section of the graph. The TNR is near 1, meaning if we predict 0, that's what is observed. We want sensitivity, the TPR, to climb toward 1, so that when we predict a 1, we observe a 1. When we are half way along the transition (τ has moved from 1.0 to 0.5), the predictions might start getting worse.

If a predictive model is not well suited to the data, the ROC curve will follow the diagonal line in the graph. This means, basically, that when the model predictions change from 0 to 1, they are not making the predictions more accurate.

The AUC value, the "area under the curve," is often thought of as a summary of how well the model fits with the data.

```
library(pROC)
bl5roc <- roc(model.frame(bushlogit5)$bushvote, logitpred)
bl5roc
```

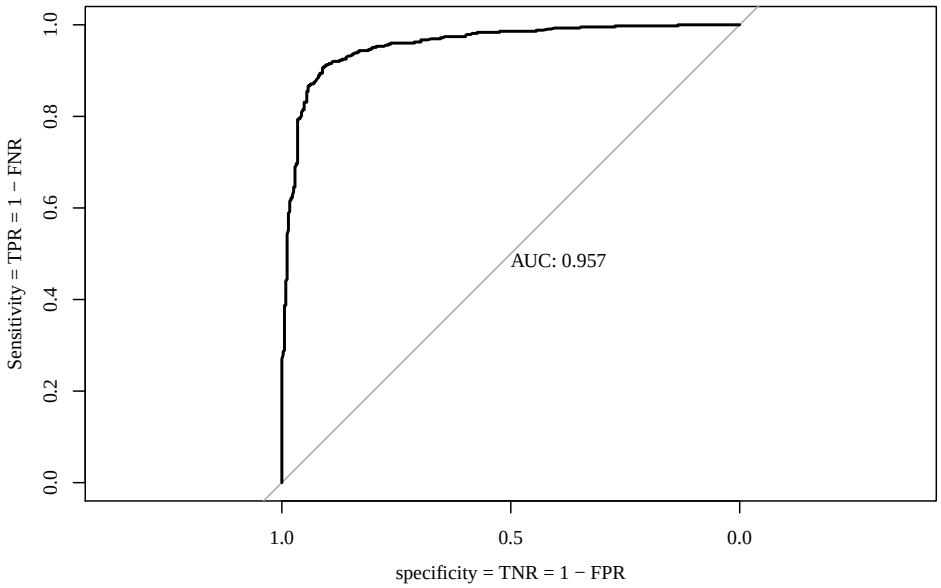
```
Call:
roc.default(response = model.frame(bushlogit5)$bushvote, predictor = logitpred)

Data: logitpred in 349 controls (model.frame(bushlogit5)$bushvote 0) < 426 cases
    (model.frame(bushlogit5)$bushvote 1).
Area under the curve: 0.957
```

References

R Core Team (2018). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria: R Foundation for Statistical Computing.

Figure 12: ROC Diagram for full model

```
plot(bl5roc, print.auc = TRUE, xlab = "specificity = TNR = 1 -
   FPR", ylab = "Sensitivity = TPR = 1 - FNR")
```



Replication Information

Please leave this next code chunk if you are producing a guide document.

```
R version 3.5.1 (2018-07-02)
Platform: x86_64-pc-linux-gnu (64-bit)
Running under: Ubuntu 18.04.1 LTS

Matrix products: default
BLAS: /usr/lib/x86_64-linux-gnu/blas/libblas.so.3.7.1
LAPACK: /usr/lib/x86_64-linux-gnu/lapack/liblapack.so.3.7.1

locale:
 [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C               LC_TIME=en_US.UTF-8
 [4] LC_COLLATE=en_US.UTF-8     LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=en_US.UTF-8       LC_NAME=C                  LC_ADDRESS=C
[10] LC_TELEPHONE=C             LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] pROC_1.13.0       rockchalk_1.8.119 memisc_0.99.14.11 MASS_7.3-50
[5] lattice_0.20-35   foreign_0.8-70

loaded via a namespace (and not attached):
 [1] zip_1.0.0          Rcpp_0.12.17       nloptr_1.0.4       cellranger_1.1.0
 [5] pillar_1.2.3       compiler_3.5.1     plyr_1.8.4         forcats_0.3.0
 [9] base64enc_0.1-3    tools_3.5.1        digest_0.6.15      lme4_1.1-17
[13] gtable_0.2.0       tibble_1.4.2       nlme_3.1-137       rlang_0.2.1
[17] Matrix_1.2-14      openxlsx_4.1.0     curl_3.2           pbivnorm_0.6.0
[21] haven_1.1.1        rio_0.5.10         repr_0.15.0        stats4_3.5.1
[25] grid_3.5.1         data.table_1.11.4  readxl_1.1.0       lavaan_0.6-1
```

```
[29] minqa_1.2.4        carData_3.0-1      ggplot2_2.2.1      car_3.0-0
[33] magrittr_1.5       scales_0.5.0       htmltools_0.3.6    kutils_1.49
[37] splines_3.5.1      abind_1.4-5        mnormt_1.5-5       colorspace_1.3-2
[41] xtable_1.8-2       lazyeval_0.2.1     munsell_0.5.0
```