Lab 6: Generalized Linear Regression

Hao Wang

March 19, 2017

1. Load essential packages

```
library(VGAM)
library(car)
library(MASS)
library(effects)
library(ggplot2)
library(Zelig)
library(ZeligChoice)
```

2. Binary Dependent Variable

When our dependent variable is binary, OLS is not applicable due to the violation of the residual variance. Generally we have two different models: probit and logit. They share the same idea: coerce the predicted y values into a continuous distribution ranging from 0 to 1.

2.1 Logistic Regression

The basic idea of logistic regression is to make a logistic transformation of our linear function.

2.1.a Logistic Function

The logistic function is

$$L(y) = \frac{1}{1 + e^{-y}}$$

and

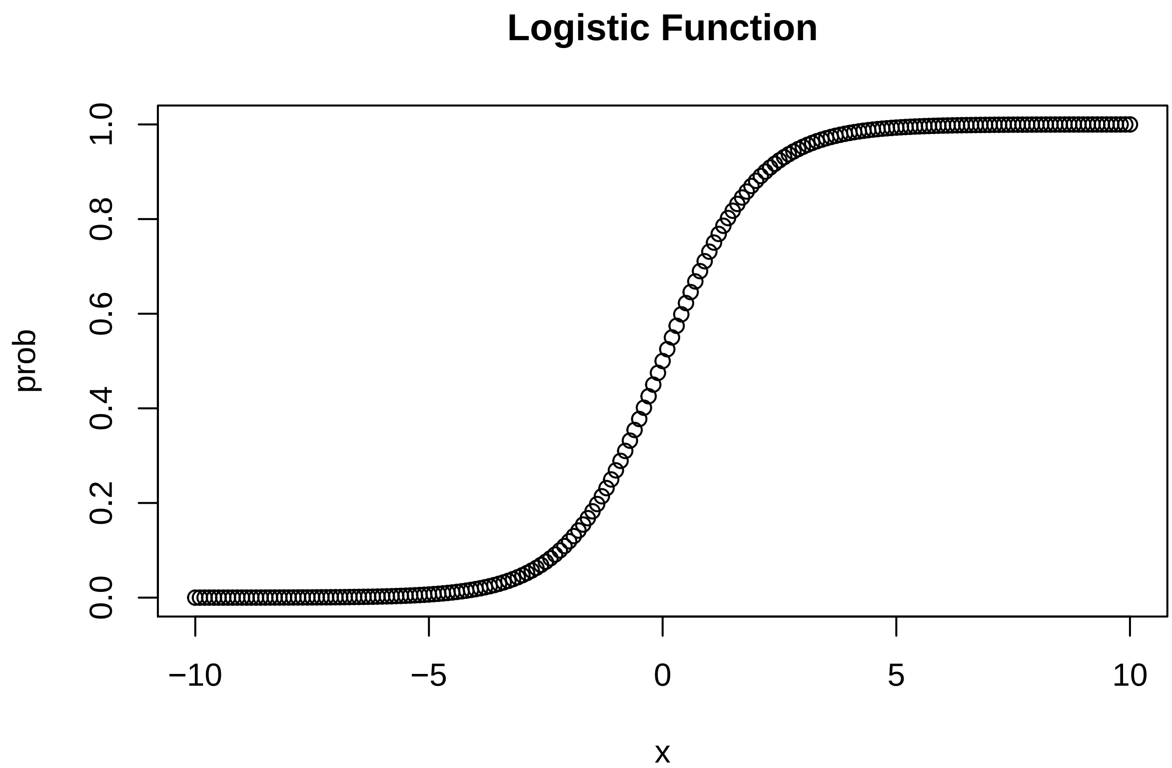
$$y = f(x) = XB = \beta_0 + \beta_1 * x_1 + \beta_2 * x_2 ...$$

Apply logistic function to the linear combination y, we got the prediction (known as the 'predicted probability' of the logistic regression)

$$Y = L(y) = L(f(x)) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 * x_1 + \beta_2 * x_2 ...)}}$$

With the logistic transformation, the outcome has a range of (0,1), we can simulate its outcome.

```
x <- seq(-10, 10, by =0.1)
prob <- 1/(1+ exp(-x))
plot(x, prob, type = "b", main = "Logistic Function")
```



2.1.b Logistic Regression R Codes

Use gender as an example, we can interpret the regression result with odds ratio. Odds ratio: the ratio of the probability the event occurs to the probability it does not occur

$$odds = \frac{p}{1-p}$$

In this example the odds ratio percentage change is -21%, which means on average the odds of being a volunteer will diminish by 21% if the subject is male.

We can also get the predicted probability by the predict fucntion.

```r
#Load Cowles data from the effects package
#Explanation
help(Cowles)
mydata <- Cowles
summary(mydata)
```

```
##   neuroticism     extraversion       sex      volunteer
##  Min.   : 0.00   Min.   : 2.00   female:780   no :824
##  1st Qu.: 8.00   1st Qu.:10.00   male  :641   yes:597
##  Median :11.00   Median :13.00
##  Mean   :11.47   Mean   :12.37
##  3rd Qu.:15.00   3rd Qu.:15.00
##  Max.   :24.00   Max.   :23.00
```

```r
logit <- glm(formula = volunteer ~ neuroticism + sex + extraversion,
             data = mydata, family = binomial(link = "logit"))
summary(logit)
```

```
##
## Call:
## glm(formula = volunteer ~ neuroticism + sex + extraversion, family = binomial(link = "logit"),
##     data = mydata)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.3977  -1.0454  -0.9084   1.2601   1.6849
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -1.116496   0.249057  -4.483 7.36e-06 ***
## neuroticism   0.006362   0.011357   0.560   0.5754
## sexmale      -0.235161   0.111185  -2.115   0.0344 *
## extraversion  0.066325   0.014260   4.651 3.30e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1933.5  on 1420  degrees of freedom
## Residual deviance: 1906.1  on 1417  degrees of freedom
## AIC: 1914.1
##
## Number of Fisher Scoring iterations: 4
```

```r
#get odds ratio
100*(exp(logit$coefficients[-1])-1)
```

```
##  neuroticism      sexmale extraversion
##     0.638215   -20.955677     6.857438
```

```r
#prediction of a certain point
new <- data.frame(sex = "male", extraversion = 10, neuroticism =10)
predict(logit, newdata = new, type = 'response')
```

```
##        1
## 0.348694
```

```r
#we can also calculate the AIC by hand
#the formula is deviance + 2*(p+1), while p is the number of variables
aic <- logit$deviance+ 2*(4)
aic
```

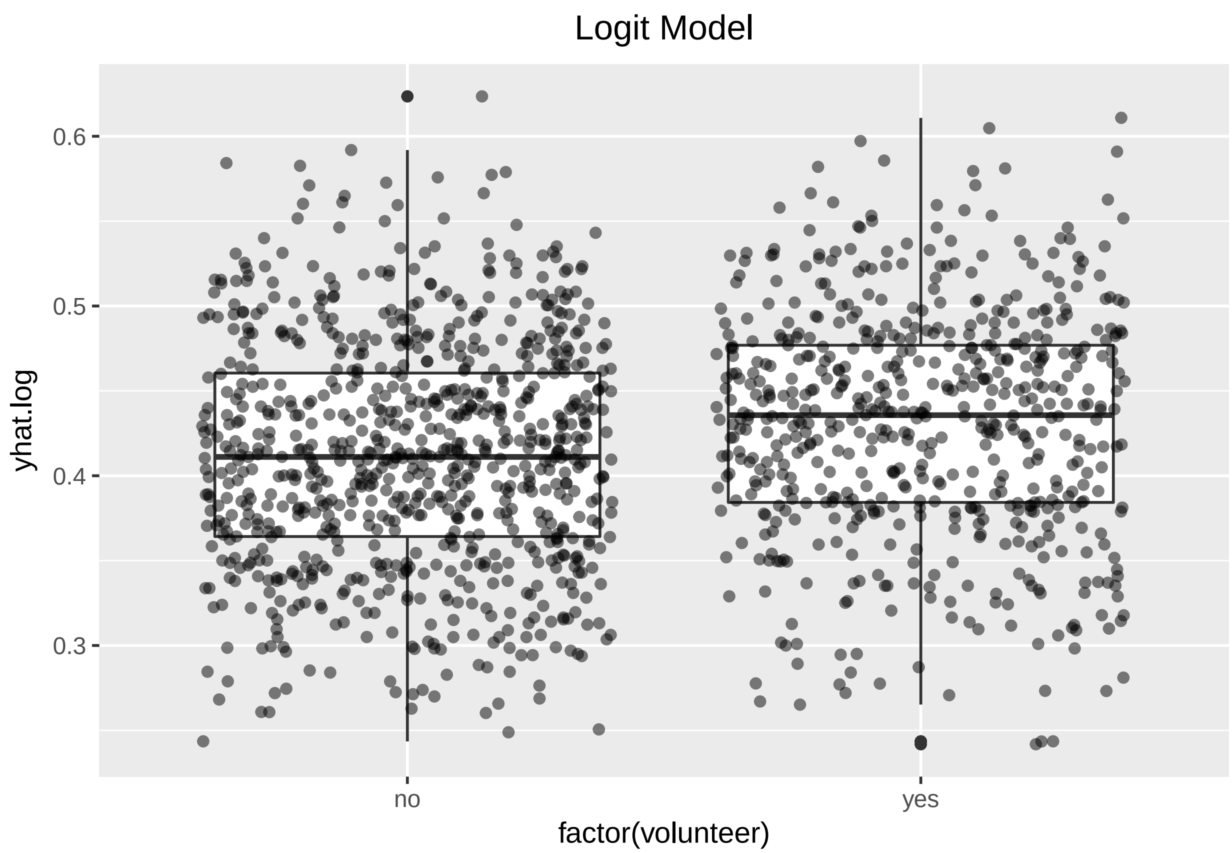
```
## [1] 1914.061
```

- Lab Practice 1: based on the logistic funtion, can you get the predicted outcome with the coefficients given by the summary table?

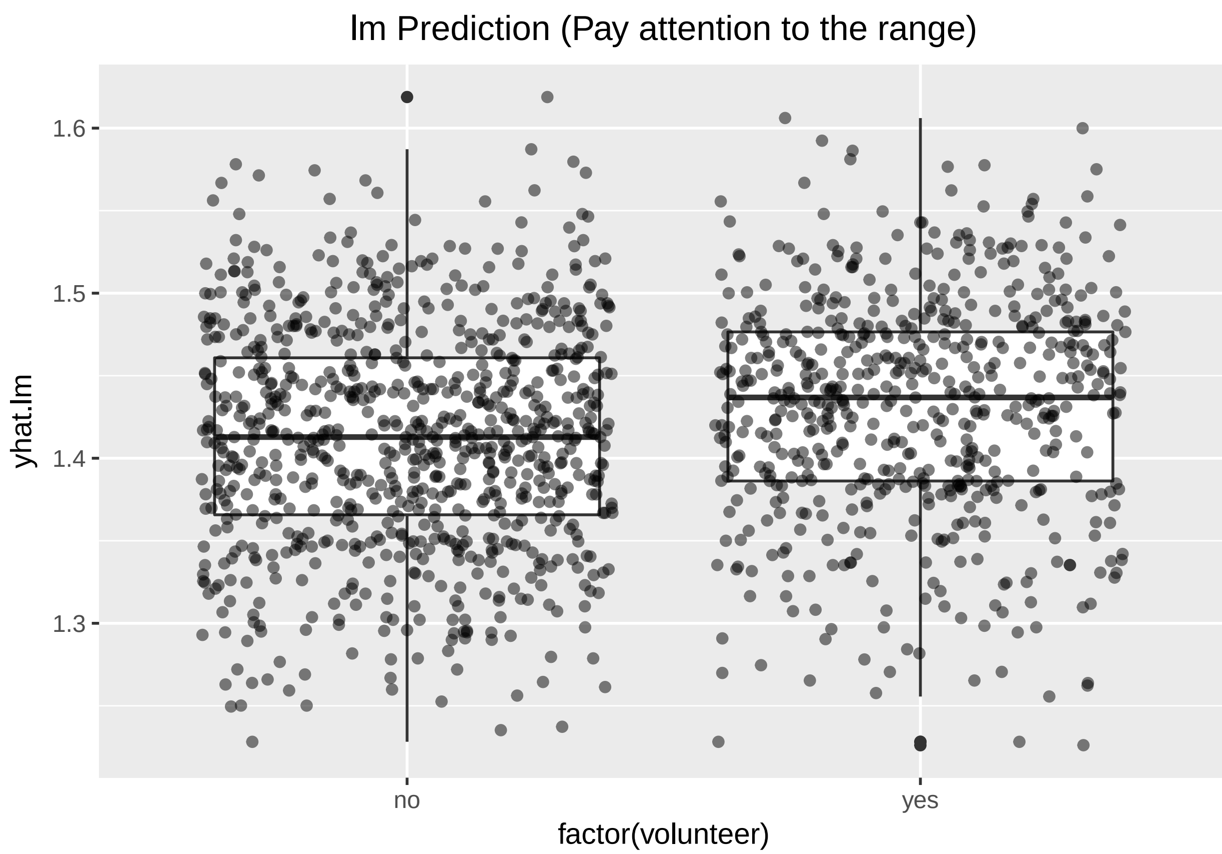
```r
#we can compute the prediction by hand
y <- logit$coefficients[1]+ logit$coefficients[2]*10 + logit$coefficients[3]*1 + logit$coefficients[4]*
#Then apply it to the logistic function
#logistic funtion is???
```

2.1.c Compare the results between logit and regular lm

```
yhat.log <- predict(logit, data=mydata, type = 'response')
lm <- lm(as.numeric(volunteer) ~ neuroticism + sex + extraversion, data = mydata)
yhat.lm <- predict(lm, data=mydata)
mydata$yhat.log <- yhat.log
mydata$yhat.lm  <- yhat.lm
ggplot(data = mydata, aes(factor(volunteer), yhat.log)) +
      geom_boxplot() +
      ggtitle("Logit Model") +
      geom_jitter(alpha = .5) +
      theme(plot.title = element_text(hjust = 0.5))
```




```
ggplot(data = mydata, aes(factor(volunteer), yhat.lm)) +
      geom_boxplot() +
      geom_jitter(alpha = .5) +
      ggtitle("lm Prediction (Pay attention to the range)") +
      theme(plot.title = element_text(hjust = 0.5))
```



2.2 Probit Model

Probit model share the similar idea with logit model, but in the probit case we are doing a gaussian transformation (the CDF funtion of a standard normal distribution).

2.2.a Probit Function

It's simply the CDF funtion of a standard normal distribution

$$\Phi(y) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{y} e^{-t^2/2} dt$$

And y is the linear combination

$$y = XB = \beta_0 + \beta_1 * x_1 + \beta_2 * x_2 ...$$

```
probit <- glm(formula = volunteer ~ neuroticism + sex + extraversion,
              data = mydata, family = binomial(link = "probit"))
summary(probit)
```

```
##
## Call:
## glm(formula = volunteer ~ neuroticism + sex + extraversion, family = binomial(link = "probit"),
##     data = mydata)
##
## Deviance Residuals:
```

```
##     Min       1Q   Median       3Q      Max  
## -1.3947  -1.0468  -0.9091   1.2611   1.6901  
## 
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)    
## (Intercept)  -0.692584   0.153670  -4.507 6.58e-06 ***
## neuroticism   0.004104   0.007044   0.583   0.5601    
## sexmale      -0.145677   0.068957  -2.113   0.0346 *  
## extraversion  0.040934   0.008790   4.657 3.21e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## (Dispersion parameter for binomial family taken to be 1)
## 
##     Null deviance: 1933.5  on 1420  degrees of freedom
## Residual deviance: 1906.1  on 1417  degrees of freedom
## AIC: 1914.1
## 
## Number of Fisher Scoring iterations: 4
```

```r
#prediction of a certain point
new <- data.frame(sex = "male", extraversion = 10, neuroticism =10)
predict(probit, newdata = new, type = 'response')
```

```
##         1 
## 0.3490514
```

- Lab Practice 2: pick another observation (you need to set values for sex, extraversion and neuroticism), compare the results of logit model and probit model.

3. Ordered Model

When our DV is an ordered/ranked outcome, such as ‘bad’, ‘ok’, ‘good’; or ‘unlikely’, ‘somewhat likely’, ‘very likely’, we need to use ordered model.

```r
mydata <- WVS
summary(mydata)
```

```
##          poverty     religion   degree          country          age       
##  Too Little :2708   no : 786   no :4238   Australia:1874   Min.   :18.00  
##  About Right:1862   yes:4595   yes:1143   Norway   :1127   1st Qu.:31.00  
##  Too Much   : 811                         Sweden   :1003   Median :43.00  
##                                           USA      :1377   Mean   :45.04  
##                                                            3rd Qu.:58.00  
##                                                            Max.   :92.00  
##     gender    
##  female:2725  
##  male  :2656  
##               
##               
##               
## 
```

```r
help(WVS)
#convert our DV to an ordered factor
```

```r
mydata$poverty <- ordered(as.factor(mydata$poverty))
#change reference level to USA
mydata$country <- relevel(mydata$country, ref="USA")
#overview
table(mydata$poverty, mydata$country)
```

```
##
##               USA Australia Norway Sweden
##   Too Little  555       952    597    604
##   About Right 372       622    494    374
##   Too Much    450       300     36     25
```

3.1 Ordered Logit

```r
# the function y~. means regressing y on the rest variables
ologit <- polr(poverty ~., method = 'logistic', data = mydata)
summary(ologit)
```

```
##
## Re-fitting to get Hessian
## Call:
## polr(formula = poverty ~ ., data = mydata, method = "logistic")
##
## Coefficients:
##                     Value Std. Error t value
## religionyes       0.17973   0.077346   2.324
## degreeyes         0.14092   0.066193   2.129
## countryAustralia -0.61778   0.070665  -8.742
## countryNorway    -0.94012   0.078468 -11.981
## countrySweden    -1.22107   0.083957 -14.544
## age               0.01114   0.001561   7.139
## gendermale        0.17637   0.052972   3.329
##
## Intercepts:
##                         Value    Std. Error t value
## Too Little|About Right   0.1120   0.1134     0.9872
## About Right|Too Much     1.9147   0.1170    16.3719
##
## Residual Deviance: 10402.59
## AIC: 10420.59
```

3.1.b Ologit Interpretation

(Also see PA book p.114) Standard interpretation of the ordered logit coefficient is that for a one unit increase in the predictor, the response variable level is expected to change by its respective regression coefficient in the ordered log-odds scale while the other variables in the model are held constant. For instance, the religious 'YES' will increase the response variable level by am ordered log-odds scale of 0.179.

We can also interpret it in terms of odds ratio. One unit increase of age, the odds that they will report 'too little work' relative to any higher categories will increase by 1.12.

```r
100*(exp((ologit$coefficients))  -1)
```

```
##      religionyes        degreeyes countryAustralia    countryNorway
##        19.689369        15.132879       -46.085741       -60.941921
##    countrySweden              age       gendermale
##       -70.508524         1.120323        19.287748
```

Prediction can be done in a similar way with logit model.

```r
new <- data.frame(country = "Norway", religion = "no", age =50, degree="no", gender="female")
predict(ologit, newdata = new, type = "prob")
```

```
##  Too Little About Right    Too Much
##  0.62129816  0.28739457  0.09130727
```

4. Introduction of Zelig Project.

Website: click here Zelig allows a unified grammar in estimating generalized linear models, besides it provides an easy way of simulating probability distributions.

4.1 Logit Model

```r
mydata <- Cowles
zlogit <- zelig(volunteer ~neuroticism + sex + extraversion, model ="logit", data=mydata)
```

```
## How to cite this model in Zelig:
##   R Core Team. 2007.
##   logit: Logistic Regression for Dichotomous Dependent Variables
##   in Christine Choirat, Christopher Gandrud, James Honaker, Kosuke Imai, Gary King, and Olivia Lau,
##   "Zelig: Everyone's Statistical Software," http://zeligproject.org/
```

```r
summary(zlogit)
```

```
## Model:
##
## Call:
## z5$zelig(formula = volunteer ~ neuroticism + sex + extraversion,
##     data = mydata)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.3977  -1.0454  -0.9084   1.2601   1.6849
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -1.116496   0.249057  -4.483 7.36e-06
## neuroticism   0.006362   0.011357   0.560   0.5754
## sexmale      -0.235161   0.111185  -2.115   0.0344
## extraversion  0.066325   0.014260   4.651 3.30e-06
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1933.5  on 1420  degrees of freedom
```

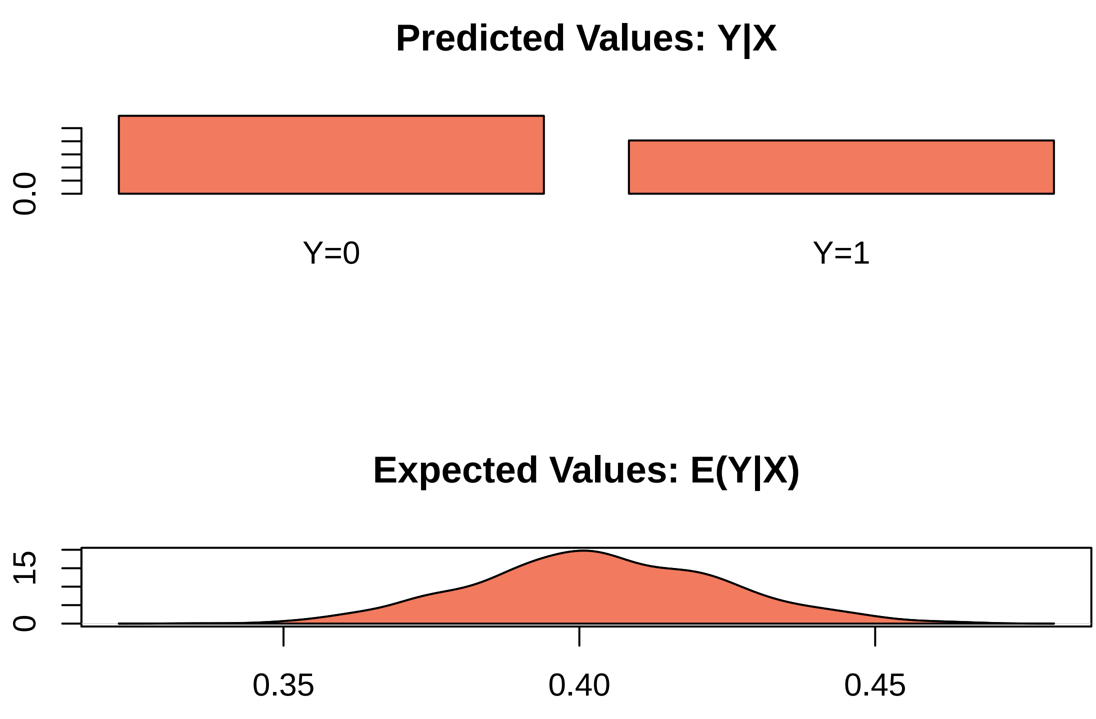
```
## Residual deviance: 1906.1  on 1417  degrees of freedom
## AIC: 1914.1
##
## Number of Fisher Scoring iterations: 4
##
## Next step: Use 'setx' method
```

We can simulate the results by sim() and setx(). (Stata package clarify uses the same command, if you're familar with Stata)

```
x.zlogit <- setx(zlogit, sex="female", neuroticism = 10, extraversion =10)
zlogit.out <- sim(zlogit, x=x.zlogit)
summary(zlogit.out)
```

```
##
##  sim x :
##  -----
## ev
##           mean         sd       50%      2.5%     97.5%
## [1,] 0.4035118 0.02088208 0.4026536 0.3630803 0.4459105
## pv
##          0     1
## [1,] 0.594 0.406
```

```
plot(zlogit.out)
```



Use the similar logit we can calculate the differences at lower level Xs and higher level Xs.

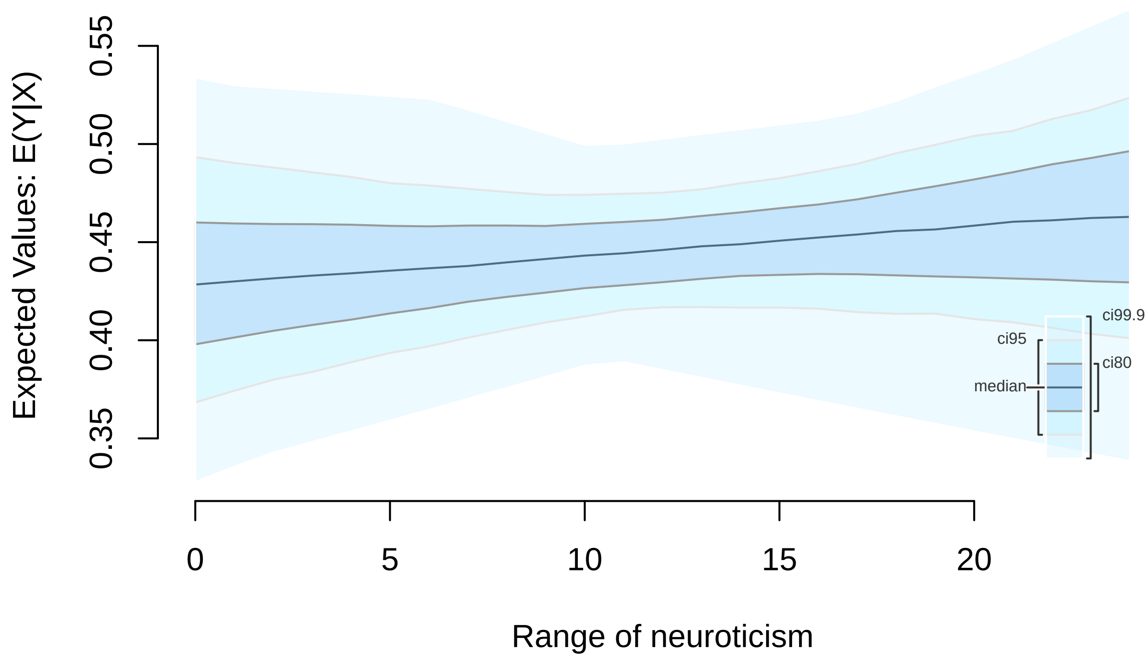
```r
#when not specified, other variables are held at mean level
x.high <- setx(zlogit, neuroticism = quantile(mydata$neuroticism, prob = 0.75), sex="male")
x.low <- setx(zlogit, neuroticism = quantile(mydata$neuroticism, prob = 0.25), sex="male")
s.out2 <- sim(zlogit, x = x.high, x1 = x.low)
summary(s.out2)
```

```
##
##  sim x :
##  -----
## ev
##           mean        sd       50%      2.5%     97.5%
## [1,] 0.3923488 0.0226508 0.3918427 0.3512686 0.4398998
## pv
##          0     1
## [1,] 0.594 0.406
##
##  sim x1 :
##  -----
## ev
##           mean        sd       50%      2.5%     97.5%
## [1,] 0.3815747 0.0203797 0.3811854 0.3427763 0.4223428
## pv
##          0     1
## [1,] 0.626 0.374
## fd
##             mean         sd         50%        2.5%      97.5%
## [1,] -0.01077412 0.01889517 -0.01093732 -0.04879821 0.02394561
```

```r
#plot(s.out2)
```

We can plot the marginal effect by setx to a sequence of values

```r
#easy way
#simulation first
#the default output of zelig is 1000 simulation at each list of x
x.sim <- setx(zlogit, neuroticism = seq(from = 0, to = 24, by = 1))
s.out3 <- sim(zlogit, x = x.sim)
plot(s.out3)
```



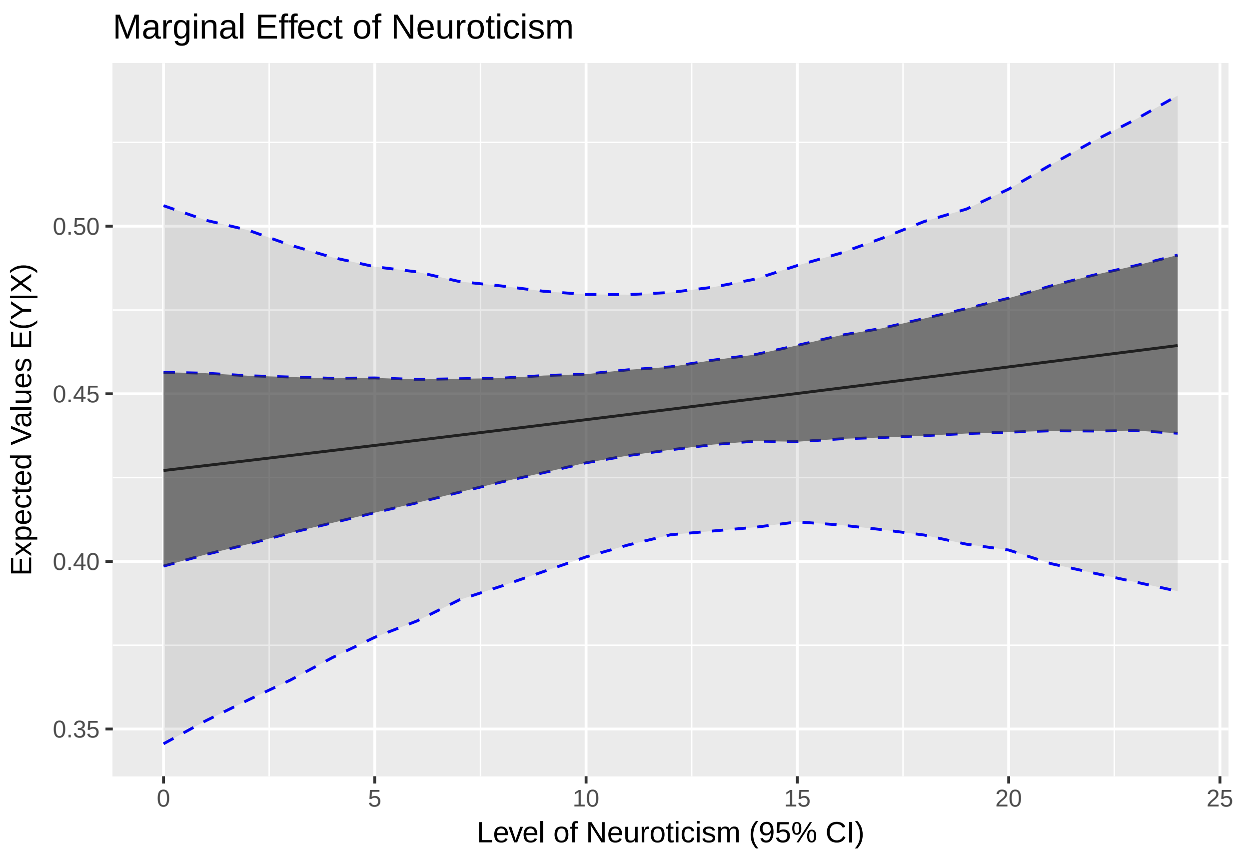

```
#`hard' way
set.seed(1)
z.out <- zelig(volunteer ~neuroticism + sex + extraversion, model ="logit", data=mydata, cite = FALSE)
neuro.range <- seq(from = 0, to = 24, by = 1)
x <- setx(z.out, neuroticism = neuro.range)
s.out <- sim(z.out, x = x)

#extract ev
myev <- s.out$get_qi(qi='ev', xvalue = 'range')
#convert the list into matrix
myev2 <- as.data.frame(matrix(unlist(myev), nrow =1000))
#create plot data
a<- apply(myev2, 2, quantile, probs = c(0.025,0.975, 0.25, 0.75))
low <- a[1,]
high <- a[2,]
qt.1 <- a[3,]
qt.3 <- a[4,]
mean <- apply(myev2, 2, mean)

plotdata <- as.data.frame(cbind(low, high, mean, qt.1, qt.3))
plotdata$neuroticism <- seq(from = 0, to = 24, by =1)


#plot in ggplot2
ggplot(data=plotdata, aes(x = neuroticism))+
  geom_line(aes(y =mean)) +
```

```
  geom_line(aes(y =high), linetype="dashed", color="blue") +
  geom_line(aes(y =low), linetype="dashed", color="blue") +
  geom_line(aes(y =qt.1), linetype="dashed", color="blue") +
  geom_line(aes(y =qt.3), linetype="dashed", color="blue") +
  xlab("Level of Neuroticism (95% CI)") +
  ylab("Expected Values E(Y|X)") +
  ggtitle("Marginal Effect of Neuroticism")+
  geom_ribbon(aes(ymin=low, ymax=high), alpha=0.1)+
  geom_ribbon(aes(ymin=qt.1, ymax=qt.3), alpha=0.6)
```



4.2 Ordered Logit Model

```
#Load Data and make sure our DV is in ordered
mydata <- WVS
mydata$poverty <- factor(mydata$poverty, ordered = TRUE, levels = c("Too Little", "About Right", "Too M
summary(mydata$poverty)
```

```
##  Too Little About Right    Too Much
##        2708        1862         811
```

```
#change reference level to USA
mydata$country <- relevel(mydata$country, ref="USA")
```

Apply the Zelig Function

```r
z.ologit <- zelig(poverty ~ country + religion + age + degree, data=mydata, model="ologit", cite = FALSE
summary(z.ologit)
```

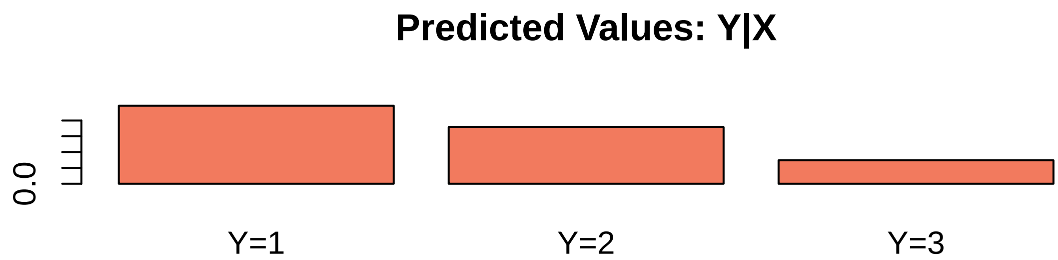
```
## Model:
## Call:
## z5$zelig(formula = poverty ~ country + religion + age + degree,
##     data = mydata)
##
## Coefficients:
##                      Value Std. Error t value
## countryAustralia -0.61939   0.070641  -8.768
## countryNorway    -0.93825   0.078449 -11.960
## countrySweden    -1.21183   0.083853 -14.452
## religionyes       0.15614   0.076982   2.028
## age               0.01128   0.001559   7.233
## degreeyes         0.13634   0.066156   2.061
##
## Intercepts:
##                         Value    Std. Error t value
## Too Little|About Right   0.0109   0.1092     0.1002
## About Right|Too Much     1.8108   0.1125    16.0919
##
## Residual Deviance: 10413.69
## AIC: 10429.69
## Next step: Use 'setx' method
```

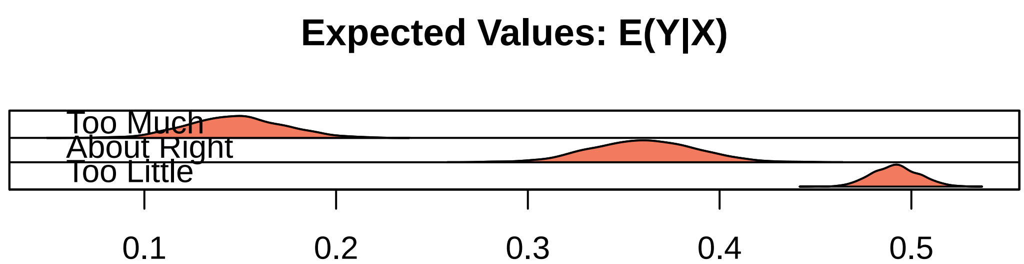
Set the explanatory variables to their observed values and simulate fitted values given x.out and view the results:

```r
x.out <- setx(z.ologit)
s.out <- sim(z.ologit, x = x.out)
summary(s.out)
```

```
##
##  sim x :
##  -----
## ev
##                   mean          sd       50%      2.5%     97.5%
## Too Little  0.4918125 0.01205533 0.4916857 0.4688848 0.5158117
## About Right 0.3603235 0.02720890 0.3603268 0.3073333 0.4135963
## Too Much    0.1478640 0.02465308 0.1475385 0.1013525 0.1967020
## pv
##       mean        sd 50% 2.5% 97.5%
## [1,] 1.654 0.7230538   2    1     3
```

```r
plot(s.out)
```





4.3 Multinomial Logit Model

Multinomial logit model should be applied when our DV contains multiple nominal categories

We load the British Election Panel Study, in which the DV is the vote choice (Conservative, Labour and Liberal Democrat)

```
#Load British Election Panel Study
mydata <- BEPS
summary(mydata$vote)
```

```
##     Conservative           Labour Liberal Democrat
##              462              720              343
```

Apply zelig function of mlogit.

In the default model, the reference cotegory is the last level and will be omitted. You can change the reference level in the way I did for ountries in the previous sections. Essentially in multinomial model our software is estimating k-1 models, while k is the number of the categories of the DV. Therefore, since the parameter estimates are relative to the referent group, the standard interpretation of the multinomial logit is that for a unit change in the predictor variable, the logit of outcome m relative to the referent group is expected to change by its respective parameter estimate (which is in log-odds units) given the variables in the model are held constant.

```
z.out <-  zelig(vote ~ age + economic.cond.national + economic.cond.household + gender + political.knowl
```

```
## How to cite this model in Zelig:
##   Thomas W. Yee. 2007.
```

```
##   mlogit: Multinomial Logistic Regression for Dependent Variables with Unordered Categorical Values
##   in Christine Choirat, Christopher Gandrud, James Honaker, Kosuke Imai, Gary King, and Olivia Lau,
##   "Zelig: Everyone's Statistical Software," http://zeligproject.org/
```

```r
summary(z.out)
```

```
## Model:
##
## Call:
## z5$zelig(formula = vote ~ age + economic.cond.national + economic.cond.household +
##     gender + political.knowledge, data = mydata)
##
##
## Pearson residuals:
##                         Min      1Q  Median     3Q   Max
## log(mu[,1]/mu[,3]) -1.911 -0.5491 -0.2180 0.8546 4.512
## log(mu[,2]/mu[,3]) -2.958 -0.7857 -0.1919 0.8202 3.551
##
## Coefficients:
##                            Estimate Std. Error z value Pr(>|z|)
## (Intercept):1              0.741085   0.423821   1.749 0.080363
## (Intercept):2             -1.078783   0.409783  -2.633 0.008474
## age:1                      0.017473   0.004734   3.691 0.000223
## age:2                     -0.002270   0.004356  -0.521 0.602293
## economic.cond.national:1  -0.451722   0.090235  -5.006 5.56e-07
## economic.cond.national:2   0.469289   0.085932   5.461 4.73e-08
## economic.cond.household:1 -0.048323   0.083399  -0.579 0.562311
## economic.cond.household:2  0.229512   0.077379   2.966 0.003016
## gendermale:1              -0.121606   0.148046  -0.821 0.411412
## gendermale:2               0.092787   0.136867   0.678 0.497813
## political.knowledge:1      0.092451   0.071525   1.293 0.196157
## political.knowledge:2     -0.272234   0.063992  -4.254 2.10e-05
##
## Number of linear predictors:  2
##
## Names of linear predictors: log(mu[,1]/mu[,3]), log(mu[,2]/mu[,3])
##
## Residual deviance: 2940.786 on 3038 degrees of freedom
##
## Log-likelihood: -1470.393 on 3038 degrees of freedom
##
## Number of iterations: 4
##
## Reference group is level  3  of the response
## Next step: Use 'setx' method
```

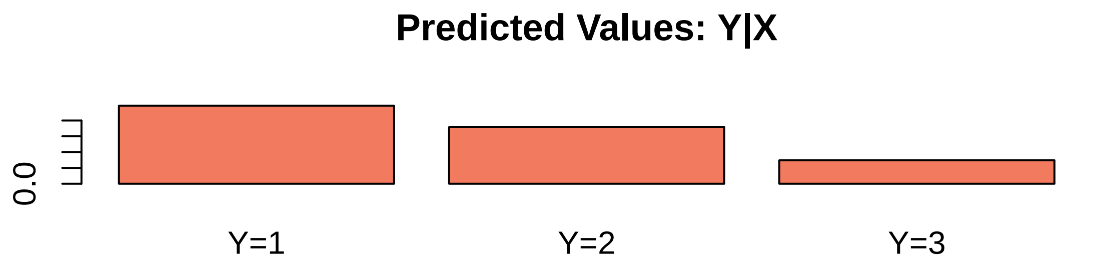
We can also simulte the results in a similar way

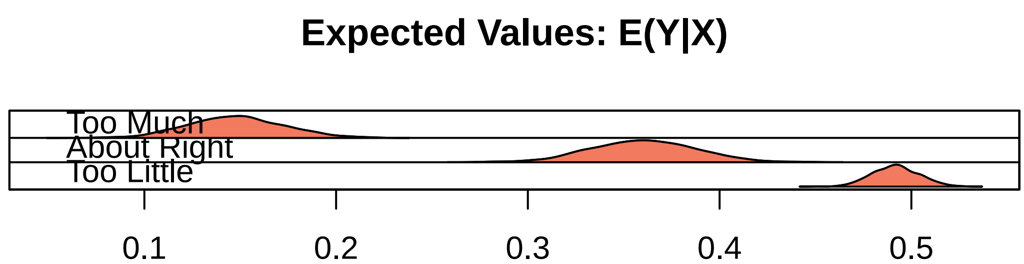
```r
x.weak <- setx(z.out, economic.cond.national = 1, economic.cond.household = 1, age =45,
               gender ="male", political.knowledge = 2)
x.strong <- setx(z.out, economic.cond.national = 4, economic.cond.household = 4, age =45,
                 gender="male", political.knowledge = 2)
s.out.mlogit <- sim(z.out, x = x.strong, x1 = x.weak)
summary(s.out.mlogit)
```

```
##
```

```
##  sim x :
##  -----
## ev
##                              mean         sd       50%      2.5%     97.5%
## Pr(Y=Conservative)     0.1368371 0.01516085 0.1363224 0.1071759 0.1688773
## Pr(Y=Labour)           0.6576552 0.02345260 0.6567823 0.6104897 0.7035967
## Pr(Y=Liberal Democrat) 0.2055077 0.01996537 0.2050475 0.1696704 0.2474367
## pv
##          1     2     3
## [1,] 0.136 0.661 0.203
##
##  sim x1 :
##  -----
## ev
##                               mean         sd        50%       2.5%
## Pr(Y=Conservative)     0.67888434 0.04701067 0.67977653 0.58251715
## Pr(Y=Labour)           0.09057999 0.01783253 0.08870294 0.06002129
## Pr(Y=Liberal Democrat) 0.23053566 0.04139992 0.22769761 0.15907650
##                            97.5%
## Pr(Y=Conservative)     0.7630633
## Pr(Y=Labour)           0.1307768
## Pr(Y=Liberal Democrat) 0.3173694
## pv
##          1     2    3
## [1,] 0.684 0.086 0.23
## fd
##                                mean         sd         50%        2.5%
## Pr(Y=Conservative)      0.54204728 0.05200626  0.54193916  0.43989656
## Pr(Y=Labour)           -0.56707525 0.03111341 -0.56735825 -0.62801777
## Pr(Y=Liberal Democrat)  0.02502797 0.04911435  0.02564651 -0.06721756
##                             97.5%
## Pr(Y=Conservative)      0.6361588
## Pr(Y=Labour)           -0.5068741
## Pr(Y=Liberal Democrat)  0.1241885
```

```r
plot(s.out)
```


Predicted Values: Y|X
0.0
Y=1
Y=2
Y=3


Expected Values: E(Y|X)
Too Much
About Right
Too Little
0.1
0.2
0.3
0.4
0.5