Problem Set 6 R Solutions

Matthew Alampay Davis

April 13, 2021

Load the data

```
amex <- read.dta13("AmEx.dta")
```

a: Transformations and summary statistics

"select" is a function in *dplyr* whose arguments are just the variables we want to keep

```
amex %<>% mutate(income = income/1000) %>% select(cardhldr, income,
    age, selfempl, ownrent, acadmos, majordrg, minordrg)
head(amex)
```

```
##   cardhldr income      age selfempl ownrent acadmos majordrg minordrg
## 1        0   14.4 27.25000        0       0       4        0        0
## 2        0   48.0 40.83333        0       1     111        0        0
## 3        1   44.0 37.66667        0       1      54        0        0
## 4        1   24.0 42.50000        0       1      60        0        0
## 5        1   35.0 21.33333        0       0       8        0        0
## 6        1   21.0 20.83333        0       0      78        0        0
```

Now calculate the fraction of consumers who are cardholders. Several ways of doing this, here's two:

```
mean(amex$cardhldr)
```

```
## [1] 0.7828467
```

```
table(amex$cardhldr) %>% prop.table
```

```
##
##         0         1
## 0.2171533 0.7828467
```

78.28% of consumers in the sample are cardholders

b: Predicting cardholder status by regression

We can continue using *lm_robust* for the linear probability model since it's just OLS. For logit and probit, we must use the *glm* command that comes in-built with R stats. To get regression output that uses robust standard errors is a bit more involved:

```
# Linear probability model
lpm.b <- lm_robust(cardhldr ~ income + age + selfempl + ownrent +
    acadmos, data = amex)
summary(lpm.b)
```

```
##
## Call:
## lm_robust(formula = cardhldr ~ income + age + selfempl + ownrent +
##     acadmos, data = amex)
```

```
##
## Standard error type:  HC2
##
## Coefficients:
##               Estimate Std. Error t value  Pr(>|t|)   CI Lower   CI Upper
## (Intercept)  6.909e-01  1.338e-02 51.6338 0.000e+00  6.647e-01  0.7171177
## income       3.897e-03  2.461e-04 15.8339 6.277e-56  3.415e-03  0.0043797
## age         -1.299e-03  4.351e-04 -2.9853 2.838e-03 -2.152e-03 -0.0004460
## selfempl    -1.020e-01  1.698e-02 -6.0079 1.931e-09 -1.353e-01 -0.0687184
## ownrent      5.010e-02  8.266e-03  6.0614 1.388e-09  3.390e-02  0.0663050
## acadmos      2.107e-05  6.135e-05  0.3434 7.313e-01 -9.919e-05  0.0001413
##                 DF
## (Intercept) 12598
## income      12598
## age         12598
## selfempl    12598
## ownrent     12598
## acadmos     12598
##
## Multiple R-squared:  0.02692 ,   Adjusted R-squared:  0.02653
## F-statistic: 83.03 on 5 and 12598 DF,  p-value: < 2.2e-16
```

```
# Logit
logit.b <- glm(cardhldr ~ income + age + selfempl + ownrent +
    acadmos, data = amex, family = binomial(link = "logit"))
coeftest(logit.b, vcov. = vcovHC, type = "HC1")
```

```
##
## z test of coefficients:
##
##                Estimate  Std. Error z value  Pr(>|z|)
## (Intercept)  0.60327591  0.08520250  7.0805 1.436e-12 ***
## income       0.02983102  0.00230752 12.9278 < 2.2e-16 ***
## age         -0.00782198  0.00248561 -3.1469   0.00165 **
## selfempl    -0.58467531  0.09084754 -6.4358 1.228e-10 ***
## ownrent      0.28944859  0.05096579  5.6793 1.353e-08 ***
## acadmos      0.00013898  0.00037732  0.3683   0.71263
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Probit
probit.b <- glm(cardhldr ~ income + age + selfempl + ownrent +
    acadmos, data = amex, family = binomial(link = "probit"))
coeftest(probit.b, vcov. = vcovHC, type = "HC1")
```

```
##
## z test of coefficients:
##
##                Estimate  Std. Error z value  Pr(>|z|)
## (Intercept)  4.1602e-01  4.8965e-02  8.4962 < 2.2e-16 ***
## income       1.5896e-02  1.2924e-03 12.2995 < 2.2e-16 ***
## age         -4.4125e-03  1.4631e-03 -3.0159  0.002563 **
## selfempl    -3.4827e-01  5.3914e-02 -6.4596  1.05e-10 ***
## ownrent      1.7386e-01  2.9391e-02  5.9154  3.31e-09 ***
## acadmos      5.7475e-05  2.1775e-04  0.2639  0.791823
## ---
```

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

c: Marginal effects at median

```
# Create an observation that's median for all covariates
# except selfselfempl = 1 and 0 Create an observation that's
# median for all covariates except ownrent = 1 and 0
median.df <- summarize_all(amex, .funs = c("median"))
new.obs <- rbind(mutate(median.df, selfempl = 1), mutate(median.df,
    selfempl = 0), mutate(median.df, ownrent = 1), mutate(median.df,
    ownrent = 0))
# Predicted probabilities
lpm.pred <- predict(lpm.b, newdata = new.obs)
logit.pred <- predict(logit.b, newdata = new.obs, type = "response")
probit.pred <- predict(probit.b, newdata = new.obs, type = "response")

# Differences
lpm.pred[1] - lpm.pred[2]
```

```
##          1
## -0.1019954
```

```
logit.pred[1] - logit.pred[2]
```

```
##          1
## -0.1222479
```

```
probit.pred[1] - probit.pred[2]
```

```
##          1
## -0.1208986
```

```
lpm.pred[3] - lpm.pred[4]
```

```
##          3
## 0.05010254
```

```
logit.pred[3] - logit.pred[4]
```

```
##          3
## 0.04910016
```

```
probit.pred[3] - probit.pred[4]
```

```
##         3
## 0.0510865
```

```
# Percentage differences
(lpm.pred[1] - lpm.pred[2])/lpm.pred[2]
```

```
##          1
## -0.1355342
```

```
(logit.pred[1] - logit.pred[2])/logit.pred[2]
```

```
##          1
## -0.1612949
```

```
(probit.pred[1] - probit.pred[2])/probit.pred[2]
```

```
##          1
## -0.1598518
```

```
(lpm.pred[3] - lpm.pred[4])/lpm.pred[4]
```

```
##         3
## 0.0665776
```

```
(logit.pred[3] - logit.pred[4])/logit.pred[4]
```

```
##          3
## 0.06478319
```

```
(probit.pred[3] - probit.pred[4])/probit.pred[4]
```

```
##          3
## 0.06754646
```

d: Marginal effects at percentiles

Income at 20th percentile:

```
# Create an observation that's median for all covariates
# except selfselfempl = 1 and 0, income = 20p Create an
# observation that's median for all covariates except ownrent
# = 1 and 0, income = 20p
median.df <- summarize_all(amex, .funs = c("median"))
new.obs <- rbind(mutate(median.df, selfempl = 1, income = quantile(amex$income,
    0.2)), mutate(median.df, selfempl = 0, income = quantile(amex$income,
    0.2)), mutate(median.df, ownrent = 1, income = quantile(amex$income,
    0.2)), mutate(median.df, ownrent = 0, income = quantile(amex$income,
    0.2)))
# Predicted probabilities
lpm.pred <- predict(lpm.b, newdata = new.obs)
logit.pred <- predict(logit.b, newdata = new.obs, type = "response")
probit.pred <- predict(probit.b, newdata = new.obs, type = "response")

# Differences
lpm.pred[1] - lpm.pred[2]
```

```
##          1
## -0.1019954
```

```
logit.pred[1] - logit.pred[2]
```

```
##          1
## -0.1316777
```

```
probit.pred[1] - probit.pred[2]
```

```
##          1
## -0.1273752
```

```
lpm.pred[3] - lpm.pred[4]
```

```
##          3
```

```
## 0.05010254
```

```
logit.pred[3] - logit.pred[4]
```

```
##          3
## 0.05494912
```

```
probit.pred[3] - probit.pred[4]
```

```
##          3
## 0.05546323
```

```
# Percentage differences
(lpm.pred[1] - lpm.pred[2])/lpm.pred[2]
```

```
##          1
## -0.1407496
```

```
(logit.pred[1] - logit.pred[2])/logit.pred[2]
```

```
##          1
## -0.1837439
```

```
(probit.pred[1] - probit.pred[2])/probit.pred[2]
```

```
##          1
## -0.1770817
```

```
(lpm.pred[3] - lpm.pred[4])/lpm.pred[4]
```

```
##          3
## 0.06913952
```

```
(logit.pred[3] - logit.pred[4])/logit.pred[4]
```

```
##          3
## 0.07667633
```

```
(probit.pred[3] - probit.pred[4])/probit.pred[4]
```

```
##        3
## 0.077107
```

Income at 80th percentile:

```
# Create an observation that's median for all covariates
# except selfselfempl = 1 and 0, income = 20p Create an
# observation that's median for all covariates except ownrent
# = 1 and 0, income = 20p
median.df <- summarize_all(amex, .funs = c("median"))
new.obs <- rbind(mutate(median.df, selfempl = 1, income = quantile(amex$income,
    0.8)), mutate(median.df, selfempl = 0, income = quantile(amex$income,
    0.8)), mutate(median.df, ownrent = 1, income = quantile(amex$income,
    0.8)), mutate(median.df, ownrent = 0, income = quantile(amex$income,
    0.8)))
# Predicted probabilities
lpm.pred <- predict(lpm.b, newdata = new.obs)
logit.pred <- predict(logit.b, newdata = new.obs, type = "response")
probit.pred <- predict(probit.b, newdata = new.obs, type = "response")
```

```r
# Differences
lpm.pred[1] - lpm.pred[2]
```

```
##          1
## -0.1019954
```

```r
logit.pred[1] - logit.pred[2]
```

```
##          1
## -0.1024671
```

```r
probit.pred[1] - probit.pred[2]
```

```
##          1
## -0.1068485
```

```r
lpm.pred[3] - lpm.pred[4]
```

```
##          3
## 0.05010254
```

```r
logit.pred[3] - logit.pred[4]
```

```
##          3
## 0.03879637
```

```r
probit.pred[3] - probit.pred[4]
```

```
##          3
## 0.04280682
```

```r
# Percentage differences
(lpm.pred[1] - lpm.pred[2])/lpm.pred[2]
```

```
##          1
## -0.1272041
```

```r
(logit.pred[1] - logit.pred[2])/logit.pred[2]
```

```
##          1
## -0.1249116
```

```r
(probit.pred[1] - probit.pred[2])/probit.pred[2]
```

```
##          1
## -0.1311442
```

```r
(lpm.pred[3] - lpm.pred[4])/lpm.pred[4]
```

```
##          3
## 0.06248566
```

```r
(logit.pred[3] - logit.pred[4])/logit.pred[4]
```

```
##          3
## 0.04729436
```

```r
(probit.pred[3] - probit.pred[4])/probit.pred[4]
```

```
##          3
## 0.05254042
```

e: Express all three cases in terms of the proability ratio or log-odds. How big ist he impact? Is this a lot or a little?

The impact is pretty substantial around -15% for being self-employed and about +7% for owning your own home. It varies for the model and is larger at the median for the nonlinear models than at the more extreme values.

f: Consider a specification that includes major and minor derogatory credit events in the last year (*majdrg*, *mindrg*). Write commands for LPM, logit and probit.

```
# Linear probability model
lpm.f <- lm_robust(cardhldr ~ income + age + selfempl + ownrent +
    acadmos + minordrg + majordrg, data = amex)
summary(lpm.f)
```

```
##
## Call:
## lm_robust(formula = cardhldr ~ income + age + selfempl + ownrent +
##     acadmos + minordrg + majordrg, data = amex)
##
## Standard error type:  HC2
##
## Coefficients:
##               Estimate Std. Error  t value   Pr(>|t|)   CI Lower
## (Intercept)  0.7120459  1.229e-02  57.9298  0.000e+00  6.880e-01
## income       0.0039274  2.272e-04  17.2899  3.253e-66  3.482e-03
## age         -0.0002363  3.951e-04  -0.5982  5.497e-01 -1.011e-03
## selfempl    -0.1004614  1.571e-02  -6.3963  1.648e-10 -1.312e-01
## ownrent      0.0455522  7.414e-03   6.1443  8.274e-10  3.102e-02
## acadmos      0.0001679  5.432e-05   3.0910  1.999e-03  6.143e-05
## minordrg    -0.0317082  5.182e-03  -6.1188  9.708e-10 -4.187e-02
## majordrg    -0.1174511  4.336e-03 -27.0883 4.094e-157 -1.260e-01
##               CI Upper    DF
## (Intercept)  0.7361392 12596
## income       0.0043727 12596
## age          0.0005381 12596
## selfempl    -0.0696749 12596
## ownrent      0.0600842 12596
## acadmos      0.0002744 12596
## minordrg    -0.0215505 12596
## majordrg    -0.1089522 12596
##
## Multiple R-squared:  0.2118 ,    Adjusted R-squared:  0.2114
## F-statistic: 225.5 on 7 and 12596 DF,  p-value: < 2.2e-16
```

```
# Logit
logit.f <- glm(cardhldr ~ income + age + selfempl + ownrent +
    acadmos + minordrg + majordrg, data = amex, family = binomial(link = "logit"))
coeftest(logit.f, vcov. = vcovHC, type = "HC1")
```

```
##
## z test of coefficients:
```

```
##
##                Estimate  Std. Error  z value  Pr(>|z|)
## (Intercept)  0.41967955  0.10668819   3.9337 8.365e-05 ***
## income       0.04519231  0.00315711  14.3145 < 2.2e-16 ***
## age          0.00106934  0.00310614   0.3443   0.73065
## selfempl    -0.76952009  0.10530994  -7.3072 2.728e-13 ***
## ownrent      0.31903274  0.05857750   5.4463 5.142e-08 ***
## acadmos      0.00198259  0.00047581   4.1667 3.090e-05 ***
## minordrg    -0.06951528  0.03596516  -1.9329   0.05325 .
## majordrg    -1.23444380  0.03737061 -33.0325 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Probit
probit.f <- glm(cardhldr ~ income + age + selfempl + ownrent +
    acadmos + minordrg + majordrg, data = amex, family = binomial(link = "probit"))
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
coeftest(probit.f, vcov. = vcovHC, type = "HC1")
```

```
##
## z test of coefficients:
##
##                Estimate  Std. Error  z value  Pr(>|z|)
## (Intercept)  0.36124137  0.06025659   5.9951 2.034e-09 ***
## income       0.02223174  0.00179182  12.4073 < 2.2e-16 ***
## age          0.00069805  0.00175431   0.3979   0.69070
## selfempl    -0.44457225  0.06049133  -7.3494 1.992e-13 ***
## ownrent      0.19109907  0.03242678   5.8932 3.787e-09 ***
## acadmos      0.00103903  0.00026429   3.9314 8.447e-05 ***
## minordrg    -0.03563930  0.02016295  -1.7676   0.07713 .
## majordrg    -0.70589892  0.02144851 -32.9113 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

g: Just see the regression outputs above

To get pseudo-R2 for probit orlogit models, we'll need the *pscl* package:

```
pR2(logit.f)
```

```
## fitting null model for pseudo-r2
```

```
##           llh       llhNull            G2      McFadden          r2ML
## -5066.3686888 -6595.4372970  3058.1372164     0.2318373     0.2154400
##          r2CU
##     0.3320299
```

Alternatively,

```
1 - (logit.f$deviance)/(logit.f$null.deviance)
```

```
## [1] 0.2318373
```

The statistic we want is the McFadden statistic here

h: Which model fits the data the best? Which variables have the most explanatory power? Do major derogatory credit events matter? Do you prefer models with or without derogatory events?

Probably model 5, the logit including the credit events. Everything but age appears to be significant.Major derogatory credit events seem to be the key explanatory variable both increasing the fit and having a substantial impact on outcomes.

i: Construct a 95% confidence interval for the effect of an additional major derogatory credit event has on being an American Express cardholder under the linear probability model.

For the LPM the marginal effect is just the confidence interval for $\beta_{majordrg}$:

```
confint(lpm.f)
```

```
##                     2.5 %        97.5 %
## (Intercept)  6.879527e-01  0.7361392109
## income       3.482178e-03  0.0043726839
## age         -1.010768e-03  0.0005380777
## selfempl    -1.312480e-01 -0.0696748752
## ownrent      3.102017e-02  0.0600842348
## acadmos      6.142583e-05  0.0002743747
## minordrg    -4.186585e-02 -0.0215505002
## majordrg    -1.259501e-01 -0.1089521636
```

j: Repeat for the logit and probit models in the case where the x values take on the median values in the sample.

We'll need the *margins* package. And recall we've already defined a data.frame that takes on median values for each covariate. The margins command requires us to convert this into list form so that's what we use for the second argument.

```
logit.se <- vcovHC(logit.f, type = "HC1")
probit.se <- vcovHC(probit.f, type = "HC1")
margins(logit.f, at = as.list.data.frame(median.df), vcov = logit.se) %>%
    summary
```

```
##    factor cardhldr  income     age selfempl ownrent acadmos majordrg
##   acadmos   1.0000 26.1550 31.5000   0.0000  0.0000 30.0000   0.0000
##       age   1.0000 26.1550 31.5000   0.0000  0.0000 30.0000   0.0000
##    income   1.0000 26.1550 31.5000   0.0000  0.0000 30.0000   0.0000
##  majordrg   1.0000 26.1550 31.5000   0.0000  0.0000 30.0000   0.0000
##  minordrg   1.0000 26.1550 31.5000   0.0000  0.0000 30.0000   0.0000
##   ownrent   1.0000 26.1550 31.5000   0.0000  0.0000 30.0000   0.0000
##  selfempl   1.0000 26.1550 31.5000   0.0000  0.0000 30.0000   0.0000
##  minordrg     AME     SE        z      p   lower   upper
##    0.0000  0.0003 0.0001   4.0974 0.0000  0.0001  0.0004
##    0.0000  0.0001 0.0004   0.3449 0.7302 -0.0007  0.0009
##    0.0000  0.0059 0.0004  14.9218 0.0000  0.0051  0.0067
##    0.0000 -0.1618 0.0054 -29.8698 0.0000 -0.1724 -0.1512
##    0.0000 -0.0091 0.0047  -1.9458 0.0517 -0.0183  0.0001
```

```
##    0.0000  0.0418 0.0083   5.0349 0.0000  0.0255  0.0581
##    0.0000 -0.1009 0.0137  -7.3835 0.0000 -0.1276 -0.0741
```

```
margins(probit.f, at = as.list.data.frame(median.df), vcov = probit.se) %>%
    summary
```

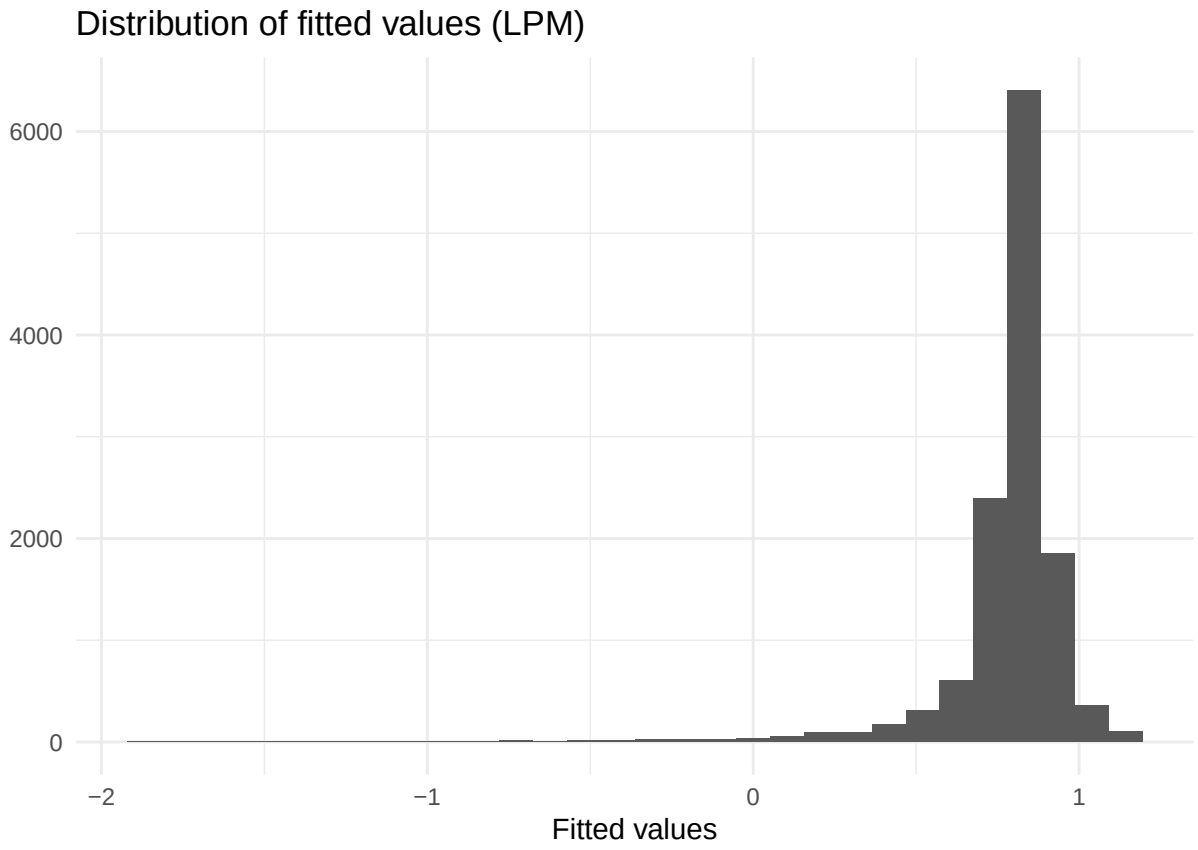
```
##    factor cardhldr  income     age selfempl ownrent acadmos majordrg
##   acadmos   1.0000 26.1550 31.5000   0.0000  0.0000 30.0000   0.0000
##       age   1.0000 26.1550 31.5000   0.0000  0.0000 30.0000   0.0000
##    income   1.0000 26.1550 31.5000   0.0000  0.0000 30.0000   0.0000
##  majordrg   1.0000 26.1550 31.5000   0.0000  0.0000 30.0000   0.0000
##  minordrg   1.0000 26.1550 31.5000   0.0000  0.0000 30.0000   0.0000
##   ownrent   1.0000 26.1550 31.5000   0.0000  0.0000 30.0000   0.0000
##  selfempl   1.0000 26.1550 31.5000   0.0000  0.0000 30.0000   0.0000
##  minordrg     AME     SE        z      p   lower   upper
##    0.0000  0.0003 0.0001   3.8826 0.0001  0.0001  0.0004
##    0.0000  0.0002 0.0004   0.3985 0.6902 -0.0007  0.0010
##    0.0000  0.0054 0.0004  12.7939 0.0000  0.0046  0.0062
##    0.0000 -0.1715 0.0055 -31.4134 0.0000 -0.1822 -0.1608
##    0.0000 -0.0087 0.0049  -1.7768 0.0756 -0.0182  0.0009
##    0.0000  0.0464 0.0084   5.4952 0.0000  0.0299  0.0630
##    0.0000 -0.1080 0.0145  -7.4333 0.0000 -0.1365 -0.0795
```

These give us exactly the Stata confidence intervals (lower and upper)

k: Compute the fitted values for the median values of the linear probability model, and plot a histogram for the fitted values of the y variable. Do the same for logit and probit models.

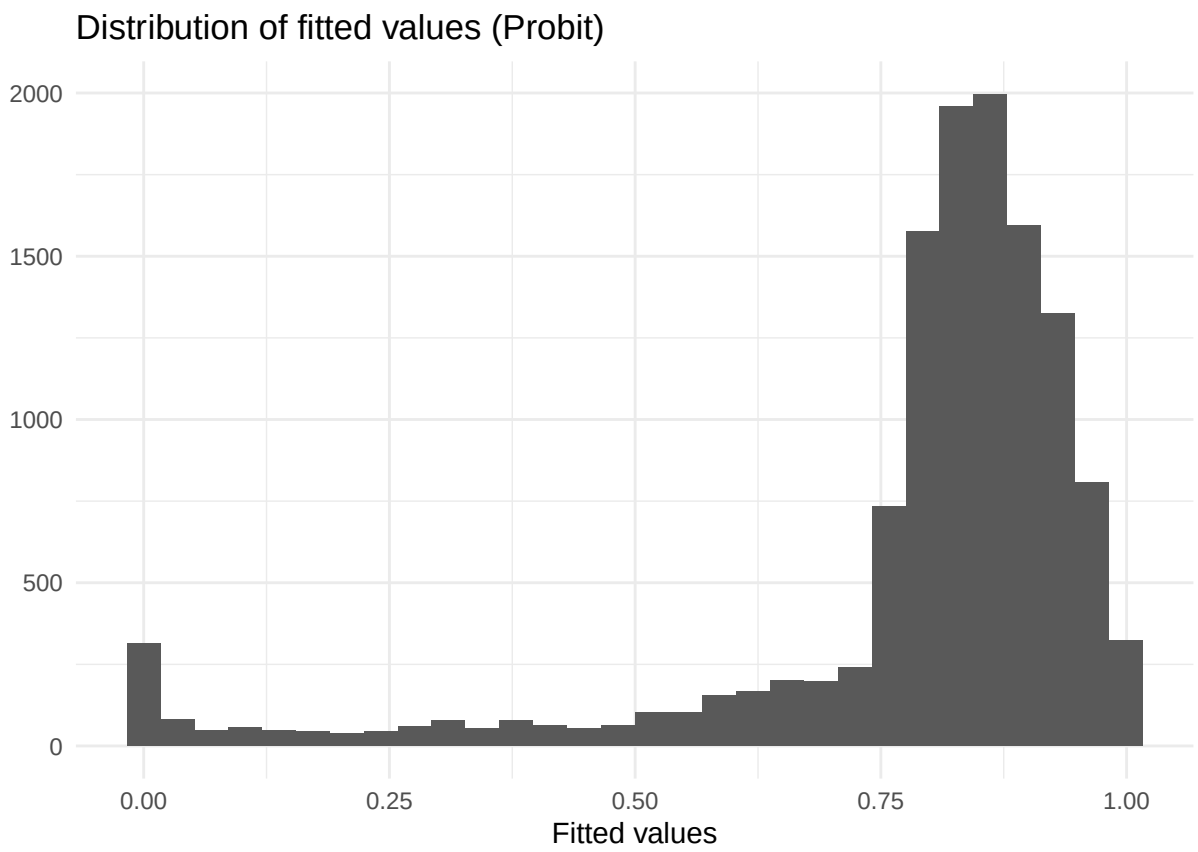
```
lpm.fit <- data.frame(y.hat = lpm.f$fitted.values)
probit.fit <- data.frame(y.hat = probit.f$fitted.values)
logit.fit <- data.frame(y.hat = logit.f$fitted.values)

ggplot(lpm.fit, aes(x = y.hat)) + theme_minimal() + ggtitle("Distribution of fitted values (LPM)") +
    ylab("") + xlab("Fitted values") + geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



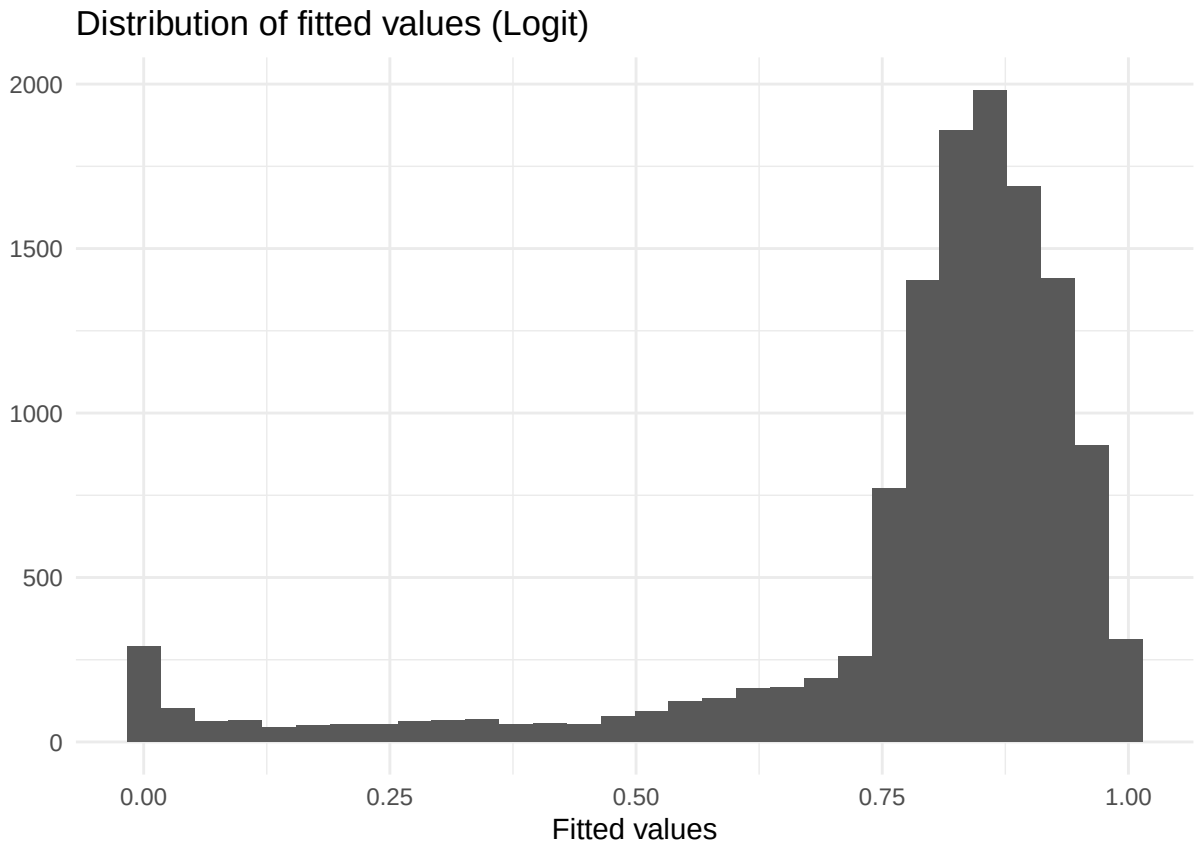

```
ggplot(probit.fit, aes(x = y.hat)) + theme_minimal() + ggtitle("Distribution of fitted values (Probit)") +
    ylab("") + xlab("Fitted values") + geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
ggplot(logit.fit, aes(x = y.hat)) + theme_minimal() + ggtitle("Distribution of fitted values (Logit)") +
    ylab("") + xlab("Fitted values") + geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



l: What should be the domain of predicted values from the LPM? What fraction of values lies outside this domain? How do we interpret them?

The domain should be between 0 and 1.

```
table(lpm.fit$y.hat < 0)
```

```
##
## FALSE  TRUE
## 12457   147
```

```
table(lpm.fit$y.hat > 1)
```

```
##
## FALSE  TRUE
## 12241   363
```

147/12604 are below zero and 363/12604 are above 1. We cannot interpret these values as probabilities. This indicates that the LPM is mis-specified, or that it is not the right tool for the job.

m: From the results you find on Table 1, what can you say about probit and logit results? How about LPM results?

The relationship between logit and probit looks okay but not the relationship to the LPM (the LPM is about half as big as it needs to be).

n: A well-known result by Amemiya (1981) is that over the range of probabilities from 30% to 70% the following approximations should hold: $\beta_{LPM} \approx 0.4\beta_{probit}$ and $\beta_{LPM} \approx 0.25\beta_{logit}$. Take the median results for logit and probit, how does this result look for the $\beta_{majordrg}$?

They do not match.