Statistics/Data Analysis

```
      name:  mainlog
       log:  C:\Users\Conor\Documents\Conor\Grad School\TA Work\Econ 103 - Econometric
> s\STATA Work\Week 10\wk10_section_log.smcl
  log type:  smcl
 opened on:  12 Mar 2018, 20:13:16

1 .
2 . // Demonstration STATA code for week 10
3 . // Principles of Econometrics 4th Edition
4 . // Covered Problems: 8.22
5 .
6 . set more off

7 . clear all

8 . use lasvegas.dta, clear

9 .
10. ///////////////////////////////////////\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
  > //////////////////////////////// Question 8.22 \\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
  > ///////////////////////////////////////\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
  >
11. ********************************************************************************
12. *Part A: Estimate the linear probability model (regression) using the model
13. * explaining DELINQUENT as a function of the remaining variables. Use the White
14. * test with cross-product terms included to test for heteroskedasticity. Why did
15. * we include the cross-product terms?
16. ********************************************************************************
17.
18. // Run OLS
19. qui reg delinquent lvr ref insur rate amount credit term arm

20.
21. // For the White test, we can use the command: estat imtest, white
22. // We are interested in the output before the table with a longer decomposition
23. estat imtest, white

  White's test for Ho: homoskedasticity
           against Ha: unrestricted heteroskedasticity

           chi2(40)      =    219.97
           Prob > chi2   =    0.0000

  Cameron & Trivedi's decomposition of IM-test

  ---------------------------------------------------
                Source |       chi2     df      p
  ---------------------+-----------------------------
    Heteroskedasticity |     219.97     40    0.0000
              Skewness |     164.83      8    0.0000
              Kurtosis |      45.66      1    0.0000
  ---------------------+-----------------------------
                 Total |     430.46     49    0.0000
  ---------------------------------------------------

24.
25. ********************************************************************************
26. *Part B: Use the estimates from (a) to estimate the error variances for each
27. * observation. How many of these estimates are at least one? How many are at
```

```
28. * most zero? How many are less than 0.01?
29. ********************************************************************************
30.
31. // Generate regression fitted values
32. predict yHat, xb

33.
34. // Generate variance using yHat*(1-yHat)
35. // Note, this variance comes from our model, and is different from eHat^2
36. gen varHat = yHat*(1-yHat)

37.
38. // Generate varHat_tab for easy presentation of the three conditions described
39. gen varHat_tab = "(3) other" // default value = "other"

40. replace varHat_tab = "(1) Less than zero" if varHat < 0
  variable varHat_tab was str9 now str18
  (135 real changes made)

41. replace varHat_tab = "(4) Greater than 1" if varHat >= 1
  (0 real changes made)

42. replace varHat_tab = "(2) Between 0 and 0.01" if varHat <= 0.01 & varHat >= 0
  variable varHat_tab was str18 now str22
  (23 real changes made)

43.
44. // Tabulate results
45. tab varHat_tab

            varHat_tab |      Freq.     Percent        Cum.
-----------------------+-----------------------------------
    (1) Less than zero |        135       13.50       13.50
  (2) Between 0 and 0.01 |       23        2.30       15.80
             (3) other |        842       84.20      100.00
-----------------------+-----------------------------------
                 Total |      1,000      100.00

46.
47. ********************************************************************************
48. *Part C: Prepare a table containing estimates and standard errors from
49. * estimating the linear probability model in each of the following ways:
50. *
51. * (i)   Least squares with conventional standard errors
52. * (ii)  Least squares with heteroskedasticity-robust standard errors.
53. * (iii) Generalized least squares omitting observations with variance less than
54. *       0.01
55. * (iv)  Generalized least squares with variances less than 0.01 changed to 0.01
56. * (v)   Generalized least squares with variances less than 0.00001 changed to
57. *       0.00001
58. *
59. * Discuss and compare the results.
60. ********************************************************************************
61.
62. // Construct weights for the GLS in 4-5
63. gen wgt_4 = 1/varHat

64. replace wgt_4 = 1/0.01 if varHat < 0.01
  (158 real changes made)
```

```
65. gen wgt_5 = 1/varHat

66. replace wgt_5 = 1/0.00001 if varHat < 0.00001
  (135 real changes made)

67.
68. // Run the regressions separately, and store results using eststo
69. qui reg delinquent lvr ref insur rate amount credit term arm

70. eststo ols

71.
72. qui reg delinquent lvr ref insur rate amount credit term arm, robust

73. eststo ols_RSE

74.
75. qui reg delinquent lvr ref insur rate amount credit term arm [aweight = 1/varHat] if
  >  varHat > 0.01

76. eststo gls_omit

77.
78. qui reg delinquent lvr ref insur rate amount credit term arm [aweight = wgt_4]

79. eststo gls_t01

80.
81. qui reg delinquent lvr ref insur rate amount credit term arm [aweight = wgt_5]

82. eststo gls_t0001

83.
84. // Prepare table for results from the 5 models
85. esttab, b(%7.4f) se(%7.4f) stats(F N) star compress mtitles

  ___________________________________________________________________________
                   (1)          (2)          (3)          (4)          (5)
                   ols      ols_RSE     gls_omit      gls_t01    gls_t0001
  ___________________________________________________________________________
  lvr           0.0016*      0.0016*      0.0016       0.0009*      0.0005*
              (0.0008)     (0.0007)     (0.0008)     (0.0004)     (0.0002)

  ref          -0.0593*     -0.0593*     -0.0571**    -0.0327*     -0.0267*
              (0.0238)     (0.0240)     (0.0211)     (0.0146)     (0.0105)

  insur        -0.4816***   -0.4816***   -0.5016***   -0.4770***   -0.5127
              (0.0236)     (0.0304)     (0.0292)     (0.0297)     (0.4086)

  rate          0.0344***    0.0344***    0.0413***    0.0204***    0.0002
              (0.0086)     (0.0098)     (0.0082)     (0.0057)     (0.0048)

  amount        0.0238       0.0238       0.0258*      0.0187      -0.0045
              (0.0127)     (0.0145)     (0.0121)     (0.0099)     (0.0089)

  credit       -0.0004*     -0.0004*     -0.0004*     -0.0002      -0.0000
              (0.0002)     (0.0002)     (0.0002)     (0.0001)     (0.0001)

  term         -0.0126***   -0.0126***   -0.0190***   -0.0065**    -0.0012
              (0.0035)     (0.0036)     (0.0041)     (0.0021)     (0.0018)

  arm           0.1283***    0.1283***    0.2089***    0.0419**     0.0188
              (0.0319)     (0.0277)     (0.0407)     (0.0140)     (0.0109)
  _cons          0.6885**     0.6885**     0.7157***    0.5587***    0.5616
               (0.2112)     (0.2285)     (0.1952)     (0.1317)     (0.4188)
  ----------------------------------------------------------------------------
  F             62.7687      43.3907      45.6690      44.9154       2.0936
  N             1.0e+03      1.0e+03     842.0000      1.0e+03      1.0e+03
  ----------------------------------------------------------------------------
  Standard errors in parentheses
  * p<0.05, ** p<0.01, *** p<0.001

86. eststo clear

87.
88. ****************************************************************************
89. *Part D: Using the results from (iv), interpret each of the coefficients.
90. * Mention whether the signs are reasonable and whether they are significantly
91. * different from zero.
92. ****************************************************************************
93.
94. reg delinquent lvr ref insur rate amount credit term arm [aweight = wgt_4]
  (sum of wgt is   2.6470e+04)

      Source |       SS           df       MS      Number of obs   =     1,000
-------------+----------------------------------   F(8, 991)       =     44.92
       Model |  12.6316713         8  1.57895891   Prob > F        =    0.0000
    Residual |  34.8377088       991  .035154096   R-squared       =    0.2661
-------------+----------------------------------   Adj R-squared   =    0.2602
       Total |    47.46938       999  .047516897   Root MSE        =    .18749

------------------------------------------------------------------------------
  delinquent |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
         lvr |   .0008577    .000379     2.26   0.024     .0001139    .0016015
         ref |  -.0326787   .0146471    -2.23   0.026    -.0614216   -.0039357
       insur |  -.4769852   .0296842   -16.07   0.000    -.5352363   -.4187342
        rate |   .0203535   .0056992     3.57   0.000     .0091697    .0315374
      amount |   .0187387    .009943     1.88   0.060     -.000773    .0382504
      credit |  -.0001617   .0001183    -1.37   0.172    -.0003939    .0000705
        term |  -.0065417   .0020826    -3.14   0.002    -.0106284   -.0024549
         arm |    .041931   .0139866     3.00   0.003     .0144843    .0693777
       _cons |   .5586635   .1316754     4.24   0.000      .300269    .8170581
------------------------------------------------------------------------------

95.
96. ****************************************************************************
97. *Appendix A - Construct test stat for White test "by hand"
98. ****************************************************************************
99.
100 // In Part A above, we just used estat imtest to calculate the test statistic
101 // for the White test of homoskedasticity of the error term. Here, I walk
102 // through conducting the test by estimating a secondary regression and then
103 // using the N*R^2 formula given in the textbook in Section 8.2.2
104
105 // use the describe command to collect names of all original variables in the
106 // dataset (i.e. lvr to delinquent)
107 qui describe lvr-delinquent, varlist

108 local origVars = r(varlist)

109 // make a local (excl) that holds name: delinquent
110 local excl delinquent
```

```
// generate a local that includes all the origVars, excluding delinquent. These
// are the RHS varibles for our main regression
local meanRHS : list origVars -excl

// Use the following loop to make variables for the square and interaction of
// all the RHS variables from the main regression. A squared value is call var_2
// and an interaction is called int_var1_var2, where var, var1, var2 are the
// names of the original variables.
local varCount : word count `meanRHS'

forvalues x = 1/`=`varCount'' {
2.         forvalues y = `x'/`=`varCount'' {
3.
                // if `x' == `y' then we have a squared term
                if `x' == `y' {
4.
                        // meanRHS is a list of variable names
                        // use local xvar : word `x' of `meanRHS' to extract the xth
> name
                        // of the list
                        local xvar : word `x' of `meanRHS'
5.                         gen `xvar'_2 = `xvar'^2
6.
                        label variable `xvar'_2 "`xvar' Squared"
7.
                }
8.                 // otherwise, we have an interaction term
                else {
9.
                        local xvar : word `x' of `meanRHS'
10.                        local yvar : word `y' of `meanRHS'
11.                        gen int_`xvar'_`yvar' = `xvar'*`yvar'
12.
                        label variable int_`xvar'_`yvar' "int `xvar' `yvar'"
13.
                } // end if x == y ... else ...
14.        } // end forvalues y
15. } // end forvalues x

// Re-run regression of delinquent on original variables
qui reg delinquent `meanRHS'

// Quietly calculate the white test value using estat imtest
qui estat imtest, white

scalar whiteTestVal_STATA = r(chi2)

// Generate residuals and square of residuals
predict eHat, residual

gen eHatSqr = eHat^2

// regress eHatSqr on level, interactions, and squares of all variables
// Given the names for the variables we built in the loop earlier, we can refer
// to all these variables using the wildcard * to get all the squared (_2) and
```

```
148 // interaction (int_) terms.
149 reg eHatSqr `meanRHS' *_2 int_*
  note: arm omitted because of collinearity
  note: ref_2 omitted because of collinearity
  note: insur_2 omitted because of collinearity
  note: arm_2 omitted because of collinearity

      Source |       SS           df       MS      Number of obs   =     1,000
-------------+----------------------------------   F(40, 959)      =      6.76
       Model |  7.51367293        40  .187841823   Prob > F        =    0.0000
    Residual |  26.6434732       959  .027782558   R-squared       =    0.2200
-------------+----------------------------------   Adj R-squared   =    0.1874
       Total |  34.1571461       999  .034191337   Root MSE        =    .16668

------------------------------------------------------------------------------
     eHatSqr |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
         lvr |  -.0042071   .0082748    -0.51   0.611    -.0204459    .0120317
         ref |  -.1426573   .2561221    -0.56   0.578    -.6452819    .3599672
       insur |  -.2074383   .2466591    -0.84   0.401    -.6914922    .2766157
        rate |  -.0921778   .0986614    -0.93   0.350    -.2857949    .1014393
      amount |   .1578815   .1651618     0.96   0.339    -.1662386    .4820017
      credit |   .0010925   .0025459     0.43   0.668    -.0039037    .0060886
        term |  -.0118508   .0394968    -0.30   0.764     -.089361    .0656593
         arm |          0  (omitted)
       lvr_2 |  -.0000269    .000024    -1.12   0.263     -.000074    .0000202
       ref_2 |          0  (omitted)
     insur_2 |          0  (omitted)
      rate_2 |   .0023687   .0022496     1.05   0.293     -.002046    .0067834
    amount_2 |   .0031948   .0032302     0.99   0.323    -.0031442    .0095338
    credit_2 |  -1.85e-07   1.41e-06    -0.13   0.896    -2.94e-06    2.57e-06
      term_2 |    .000088   .0004571     0.19   0.847     -.000809    .0009851
       arm_2 |          0  (omitted)
 int_lvr_ref |   .0015055   .0011384     1.32   0.186    -.0007286    .0037396
int_lvr_insur |  .0006097    .000968     0.63   0.529      -.00129    .0025094
int_lvr_rate |   .0001351    .000391     0.35   0.730    -.0006321    .0009024
int_lvr_amount | -.0000102  .0005779    -0.02   0.986    -.0011443     .001124
int_lvr_credit | -7.10e-06  8.37e-06    -0.85   0.396    -.0000235    9.33e-06
int_lvr_term |    .000346   .0001296     2.67   0.008     .0000917    .0006003
 int_lvr_arm |  -.0002256   .0014284    -0.16   0.875    -.0030288    .0025777
int_ref_insur |  -.0152041  .0279869    -0.54   0.587    -.0701268    .0397186
int_ref_rate |   .0128034   .0103218     1.24   0.215    -.0074526    .0330594
int_ref_amount |  .0147192  .0156202     0.94   0.346    -.0159346    .0453729
int_ref_credit |  .0000464  .0002568     0.18   0.857    -.0004576    .0005504
int_ref_term |  -.0045631   .0053209    -0.86   0.391     -.015005    .0058788
 int_ref_arm |   .0173632   .0379412     0.46   0.647    -.0570941    .0918206
int_insur_rate |  .0175694  .0102066     1.72   0.086    -.0024604    .0375991
int_insur_amount | .0070806 .0155683     0.45   0.649    -.0234711    .0376324
int_insur_credit | .0000168 .0002321     0.07   0.942    -.0004388    .0004724
int_insur_term | -.0046649   .004191    -1.11   0.266    -.0128894    .0035596
int_insur_arm | -.0203728   .0402332    -0.51   0.613     -.099328    .0585823
int_rate_amount | -.008355  .0061225    -1.36   0.173    -.0203701    .0036601
int_rate_credit | -9.72e-06 .0000896    -0.11   0.914    -.0001856    .0001662
int_rate_term |  .0014639   .0016591     0.88   0.378    -.0017919    .0047197
int_rate_arm |   .0287677   .0162178     1.77   0.076    -.0030588    .0605941
int_amount_credit | .000168 .0001258     1.34   0.182    -.0000788    .0004148
int_amount_term | -.0076087  .004146    -1.84   0.067     -.015745    .0005275
int_amount_arm | .0059363   .0302796     0.20   0.845    -.0534857    .0653582
int_credit_term | -.0000199  .000035    -0.57   0.569    -.0000887    .0000488
int_credit_arm | -.0000659  .0003403    -0.19   0.846    -.0007338     .000602
int_term_arm |  -.0040533   .0101548    -0.40   0.690    -.0239815    .0158749
       _cons |   .4868574   1.408049     0.35   0.730    -2.276356    3.250071
------------------------------------------------------------------------------
```

```

// Note that ref_2, insur_2, and arm_2, along with arm, are all dropped due to
// collinearity. This reduces our 44 RHS terms (not including the constant)
// down to 40 RHS terms. This matches the "model degrees of freedom" in STATA,
// stored as e(df_m)

// Calculate white test statistic as R^2 * N
scalar whiteTestVal_hand = e(r2)*e(N)


// Compare the "hand" calculated whiteTestVal to that given by STATA:
disp "Hand calculated White test statistic: " whiteTestVal_hand
  Hand calculated White test statistic: 219.97367

disp "STATA-calculated White test statistic: " whiteTestVal_STATA
  STATA-calculated White test statistic: 219.97368


// Calculate the 99.5% critical value for the White test
scalar alpha = 0.005

disp "White Test " (1-alpha)*100 "% critical value: " invchi2(e(df_m),1-alpha)
  White Test 99.5% critical value: 66.765962


// Can also use STATA's built in Breusch-Pagan test to run the White Test when
// using some extra options (i.e. feeding in the RHS variables and specifying
// the iid option)
qui reg delinquent `meanRHS'

estat hettest `meanRHS' *_2 int_*, iid

  Breusch-Pagan / Cook-Weisberg test for heteroskedasticity
           Ho: Constant variance
           Variables: lvr ref insur rate amount credit term arm lvr_2 ref_2 insur_2
                      rate_2 amount_2 credit_2 term_2 arm_2 int_lvr_ref int_lvr_insur
                      int_lvr_rate int_lvr_amount int_lvr_credit int_lvr_term
                      int_lvr_arm int_ref_insur int_ref_rate int_ref_amount
                      int_ref_credit int_ref_term int_ref_arm int_insur_rate
                      int_insur_amount int_insur_credit int_insur_term int_insur_arm
                      int_rate_amount int_rate_credit int_rate_term int_rate_arm
                      int_amount_credit int_amount_term int_amount_arm int_credit_term
                      int_credit_arm int_term_arm

           chi2(40)      =   219.97
           Prob > chi2  =   0.0000


*****************************************************************************
*Appendix B - Weighted regression as OLS of transformed variables
*****************************************************************************

// Another way to calculate the values from GLS (weighted regression) is to run
// an unweighted regression on a set of transformed variables.
// Specifically, whatever you assign as the aweight in the reg command we are
// multiplying both the left- and right-hand side variables by the square root
// of that variable and dividing by the square root of the average of that
// variable, one observation at a time.
//
// Below, we walk through implementing the regression with "transformed" values
// and compare the result to running the weighted regression.

// Generate the "transformed" variables by multiplying by sqrt(wgt_4) and
// dividing by sqrt(avg_wgt_4). We call these values var_adju. In addition, we
// try the variables without dividing through by sqrt(avg_wgt_4), which we will
// compare below. The variables without dividing by sqrt(avg_wgt_4) are called
// var_adju2
qui sum wgt_4

scalar avg_wgt_4 = r(mean)

foreach x of local origVars {
  2.         gen `x'_adju = `x'*sqrt(wgt_4)/sqrt(avg_wgt_4)
  3.         gen `x'_adju2 = `x'*sqrt(wgt_4)
  4. }


// In addition, we need to calculate the "transformed" constant (which is no
// longer constant)
gen cnst_adju = sqrt(wgt_4)/sqrt(avg_wgt_4)

gen cnst_adju2 = sqrt(wgt_4)


// Construct a varlist of "adjusted" variables, and then remove
// delinquent_adju from the list, giving us the RHS variables meanRHS_adju
qui desc *_adju, varlist

local adjuVars = r(varlist)

local excl delinquent_adju

local meanRHS_adju : list adjuVars -excl


// Do the same thing, but this time collecting the _adju2 variables
qui desc *_adju2, varlist

local adju2Vars = r(varlist)

local excl delinquent_adju2

local meanRHS_adju2 : list adju2Vars -excl


// Run OLS for delinquent_adju on the adjusted RHS values
// Be sure to not let STATA include a constant in the regression! We already
// have the "transformed constant" included in the list meanRHS_adju
reg delinquent_adju `meanRHS_adju', noconstant

      Source |       SS           df       MS      Number of obs   =     1,000
-------------+----------------------------------   F(9, 991)       =     47.82
       Model |  15.1282702         9  1.68091891   Prob > F        =    0.0000
    Residual |  34.8377089       991  .035154096   R-squared       =    0.3028
-------------+----------------------------------   Adj R-squared   =    0.2964
       Total |  49.9659791     1,000  .049965979   Root MSE        =    .18749

------------------------------------------------------------------------------
delinquent~u |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
    lvr_adju |   .0008577    .000379     2.26   0.024     .0001139    .0016015
    ref_adju |  -.0326787   .0146471    -2.23   0.026    -.0614216   -.0039357
  insur_adju |  -.4769852   .0296842   -16.07   0.000    -.5352363   -.4187342
   rate_adju |   .0203535   .0056992     3.57   0.000     .0091697    .0315374
 amount_adju |   .0187387    .009943     1.88   0.060     -.000773    .0382504
 credit_adju |  -.0001617   .0001183    -1.37   0.172    -.0003939    .0000705
   term_adju |  -.0065417   .0020826    -3.14   0.002    -.0106284   -.0024549
    arm_adju |    .041931   .0139866     3.00   0.003     .0144843    .0693777
   cnst_adju |   .5586635   .1316754     4.24   0.000     .3002689    .8170581
------------------------------------------------------------------------------
```

```

// Collect RMSE and SSE to compare to the _adju2 regression below
scalar rmse_adju = e(rmse)

scalar sse_adju = e(rss)


// Do the F-test that the all terms besides beta for cnst_adju are zero:
local excl cnst_adju

local meanRHS_adju_nocnst : list meanRHS_adju -excl

test `meanRHS_adju_nocnst'

 ( 1)  lvr_adju = 0
 ( 2)  ref_adju = 0
 ( 3)  insur_adju = 0
 ( 4)  rate_adju = 0
 ( 5)  amount_adju = 0
 ( 6)  credit_adju = 0
 ( 7)  term_adju = 0
 ( 8)  arm_adju = 0

       F(  8,   991) =   44.92
            Prob > F =    0.0000


// Compare the OLS of the x*sqrt(wgt_4)/sqrt(avg_wgt_4) values to weighted
// regression with weights of wgt_4
reg delinquent `meanRHS' [aweight = wgt_4]
(sum of wgt is   2.6470e+04)

      Source |       SS           df       MS      Number of obs   =     1,000
-------------+----------------------------------   F(8, 991)       =     44.92
       Model |  12.6316713         8  1.57895891   Prob > F        =    0.0000
    Residual |  34.8377088       991  .035154096   R-squared       =    0.2661
-------------+----------------------------------   Adj R-squared   =    0.2602
       Total |    47.46938       999  .047516897   Root MSE        =    .18749

------------------------------------------------------------------------------
  delinquent |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
         lvr |   .0008577    .000379     2.26   0.024     .0001139    .0016015
         ref |  -.0326787   .0146471    -2.23   0.026    -.0614216   -.0039357
       insur |  -.4769852   .0296842   -16.07   0.000    -.5352363   -.4187342
        rate |   .0203535   .0056992     3.57   0.000     .0091697    .0315374
      amount |   .0187387    .009943     1.88   0.060     -.000773    .0382504
      credit |  -.0001617   .0001183    -1.37   0.172    -.0003939    .0000705
        term |  -.0065417   .0020826    -3.14   0.002    -.0106284   -.0024549
         arm |    .041931   .0139866     3.00   0.003     .0144843    .0693777
       _cons |   .5586635   .1316754     4.24   0.000      .300269    .8170581
------------------------------------------------------------------------------


gen del_wt = delinquent*wgt_4


// Quirk (1) - the "F-test of the regression" differs because the "transformed"
// OLS also tests the cnst_adju term, while the weighted regression does not.
// Instead, compare the "F-test" reported by the weighted regression to the
// test of the non-constant terms from the "transformed" OLS.
//
// Quirk (2) - the "Adjusted-R2" is different because of how STATA calculates
// the adjustment when the noconstant option is used. In STATA, adjusted R2 uses
// Adju R^2 = 1 - (SSE/(N-K))/(SST/(N-C)) where C = 1 if there is a constant in
// the regression and C=0 otherwise.

// Finally, lets look at what happens when we use OLS on the "adju2" variables
// i.e. x*sqrt(wgt_4) instead of x*sqrt(wgt_4)/sqrt(avg_wgt_4)
reg delinquent_adju2 `meanRHS_adju2', noconstant

      Source |       SS           df       MS      Number of obs   =     1,000
-------------+----------------------------------   F(9, 991)       =     47.82
       Model |  400.450981         9  44.4945534   Prob > F        =    0.0000
    Residual |  922.167194       991  .930542072   R-squared       =    0.3028
-------------+----------------------------------   Adj R-squared   =    0.2964
       Total |  1322.61817     1,000  1.32261817   Root MSE        =    .96465

------------------------------------------------------------------------------
delinquent~2 |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
   lvr_adju2 |   .0008577    .000379     2.26   0.024     .0001139    .0016015
   ref_adju2 |  -.0326787   .0146471    -2.23   0.026    -.0614216   -.0039357
 insur_adju2 |  -.4769852   .0296842   -16.07   0.000    -.5352363   -.4187342
  rate_adju2 |   .0203535   .0056992     3.57   0.000     .0091697    .0315374
amount_adju2 |   .0187387    .009943     1.88   0.060     -.000773    .0382504
credit_adju2 |  -.0001617   .0001183    -1.37   0.172    -.0003939    .0000705
  term_adju2 |  -.0065417   .0020826    -3.14   0.002    -.0106284   -.0024549
   arm_adju2 |    .041931   .0139866     3.00   0.003     .0144843    .0693777
  cnst_adju2 |   .5586635   .1316754     4.24   0.000      .300269    .8170581
------------------------------------------------------------------------------


// Collect the RMSE and SSE values
scalar rmse_adju2 = e(rmse)

scalar sse_adju2 = e(rss)


// The ratio of sse_adju2 to see_adju matches avg_wgt_4, and the ratio of
// rmse_adju2 to rmse_adju matches sqrt(avg_wgt_4)
disp "SSE ratio:   " sse_adju2/sse_adju "   v.s. Average of wgt_4: " avg_wgt_4
  SSE ratio:   26.470374   v.s. Average of wgt_4: 26.470374

disp "RMSE ratio: " rmse_adju2/rmse_adju "   v.s. Sqrt of Average of wgt_4:  " sqrt(
  > avg_wgt_4)
  RMSE ratio: 5.1449368   v.s. Sqrt of Average of wgt_4:  5.1449368


*****************************************************************************
*Appendix C - Relationship between variance estimate and eHatSqr
*****************************************************************************

gen val001 = 0.01

gen val000001 = 0.00001


label variable varHat "Estimated Variance"
label variable eHatSqr "Squared Error Term"

twoway (scatter varHat eHatSqr if delinquent == 0, msymbol(Oh) legend(label(1 "Delin
> quent = 0"))) ///
>                        (scatter varHat eHatSqr if delinquent == 1, msymbol(th) lege
> nd(label(2 "Delinquent = 1"))) ///
>                        (line val001 eHatSqr, legend(label(3 "VarHat = 0.01"))) ///
>                        (line val000001 eHatSqr, legend(label(4 "VarHat = 0.00001"))
> ), ///
>                        xtitle("Squared Error Term") ytitle("Estimated Variance")

//Convert log file (smcl) to pdf
```