# Using Stata dynamic tags in a text file with the dyndoc command

Let us consider an example where we study the **mpg** and **weight** variables in **auto.dta**. In our examples below, we will first write the commands so that they will be displayed in our target HTML file. Then, we will write the commands so that Stata will process the Stata dynamic tags, displaying the results of the Stata commands in the target HTML file. We first use the **sysuse** command to load the dataset and then describe the data using the **describe** command.

<<dd\_do>>  
sysuse auto, clear  
describe  
<</dd\_do>>

This produces the following Stata results:

. sysuse auto, clear  
(1978 Automobile Data)  
  
. describe  
  
Contains data from C:\Program Files (x86)\Stata15\ado\base/a/auto.dta  
 obs: 74 1978 Automobile Data  
 vars: 12 13 Apr 2016 17:45  
 size: 3,182 (\_dta has notes)  
--------------------------------------------------------------------------------  
 storage display value  
variable name type format label variable label  
--------------------------------------------------------------------------------  
make str18 %-18s Make and Model  
price int %8.0gc Price  
mpg int %8.0g Mileage (mpg)  
rep78 int %8.0g Repair Record 1978  
headroom float %6.1f Headroom (in.)  
trunk int %8.0g Trunk space (cu. ft.)  
weight int %8.0gc Weight (lbs.)  
length int %8.0g Length (in.)  
turn int %8.0g Turn Circle (ft.)  
displacement int %8.0g Displacement (cu. in.)  
gear\_ratio float %6.2f Gear Ratio  
foreign byte %8.0g origin Car type  
--------------------------------------------------------------------------------  
Sorted by: foreign

Now, we want to check if **mpg** is always greater than 0 and less than 100. We use the **assert** command to perform the check. In this case, we do not want to include any output in the target HTML file, so we use the **quietly** attribute to modify the behavior of the **dd\_do** Stata dynamic tag.

<<dd\_do:quietly>>  
assert mpg > 0 & mpg < 100  
<</dd\_do>>

If the data do not satisfy the conditions, **dyndoc** will fail with an error message, which will occur if we run the same **assert** command in a do-file. Next, we want to summarize the **weight** variable:

<<dd\_do>>  
summarize weight  
<</dd\_do>>

This produces the following in the target HTML file:

. summarize weight  
  
 Variable | Obs Mean Std. Dev. Min Max  
-------------+---------------------------------------------------------  
 weight | 74 3019.459 777.1936 1760 4840

We want to use the minimum and maximum values of **weight** in a sentence. Instead of copying and pasting the numbers from the **summarize** output, we can use the **dd\_display** Stata dynamic tag with the **r(min)** and **r(max)** stored results:

The variable weight has minimum value <<dd\_display: %4.2f `r(min)'>> and  
has maximum value <<dd\_display: %4.2f `r(max)'>>.

This produces the following in the target HTML file:

> The variable weight has minimum value 1760.00  
and has maximum value 4840.00.

The **dd\_display** dynamic tag uses Stata's **display** command to evaluate expressions. It can be used as a calculator. For example, if we want to include the

in a sentence, instead of calculating the number and then copying and pasting it, we can use

The variable weight has range <<dd\_display: %4.2f `r(max)'-`r(min)'>>.

which produces the following in the target HTML file:

> The variable weight has range 3080.00.

Now, we want to graph **mpg** and **weight** using a scatterplot. We use the **dd\_do** tag with the **nooutput** attribute to generate the scatterplot first. The **nooutput** attribute leaves the command in the output only,

<<dd\_do:nooutput>>  
scatter mpg weight, mcolor(blue%50)  
<</dd\_do>>

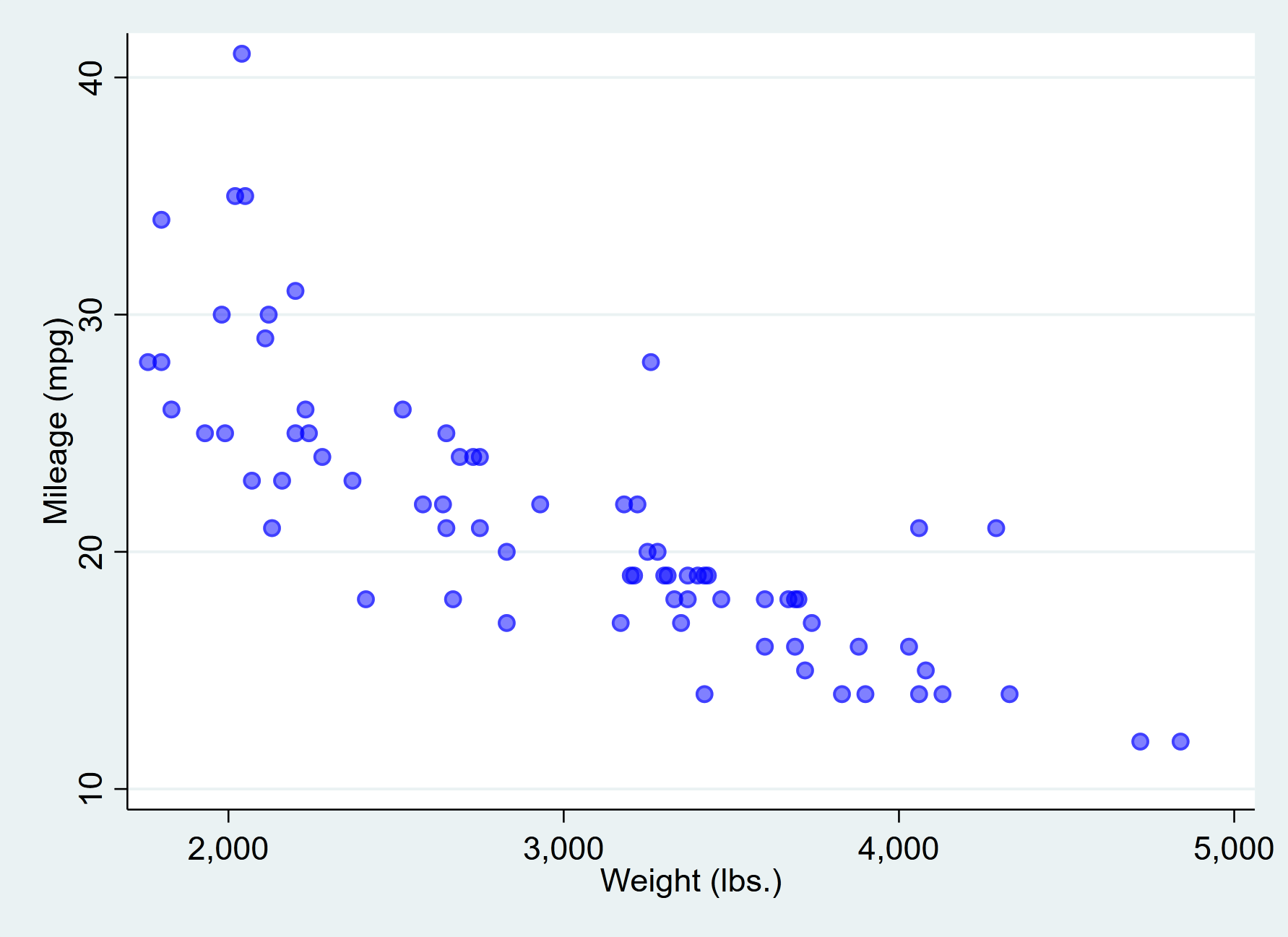
which generates a scatterplot of **mpg** and **weight** with 50% opacity color markers.

. scatter mpg weight, mcolor(blue%50)

Now, we want to export the graph to a file and include an image link to the file.

<<dd\_graph: sav("graph.png") alt("scatter mpg price") replace markdown>>

This produces following graph.



scatter mpg price

Now, we perform a linear regression between **mpg** and **weight**.

. regress mpg weight  
  
 Source | SS df MS Number of obs = 74  
-------------+---------------------------------- F(1, 72) = 134.62  
 Model | 1591.9902 1 1591.9902 Prob > F = 0.0000  
 Residual | 851.469256 72 11.8259619 R-squared = 0.6515  
-------------+---------------------------------- Adj R-squared = 0.6467  
 Total | 2443.45946 73 33.4720474 Root MSE = 3.4389  
  
------------------------------------------------------------------------------  
 mpg | Coef. Std. Err. t P>|t| [95% Conf. Interval]  
-------------+----------------------------------------------------------------  
 weight | -.0060087 .0005179 -11.60 0.000 -.0070411 -.0049763  
 \_cons | 39.44028 1.614003 24.44 0.000 36.22283 42.65774  
------------------------------------------------------------------------------

The regression shows that for every unit increase in **weight**, a -0.0060 unit increase in **mpg** is predicted.