Title

stata.com

Example 2 — Table of medians and rank-sum test results

Description Remarks and examples Reference Also see

Description

In this example, we demonstrate how to use `table` to compute medians and store them in a collection. We also use `collect` to store the results of rank-sum tests in the collection and then create a customized table combining the results.

Remarks and examples

stata.com

Remarks are presented under the following headings:

Computing and collecting statistics

Below, we use data from the Second National Health and Nutrition Examination Survey (NHANES II) (McDowell et al. 1981). We wish to compute the median `age`, `weight`, systolic blood pressure (`bpsystol`), cholesterol, and iron for individuals who have diabetes and those who do not. We use the `table` command to compute these statistics. The first set of parentheses places the variables on the rows of the table, and the second set places the levels of `diabetes` on the columns. By default, `table` will display the table and store the results in a collection called `Table`. Also by default, `table` will report the statistics for each group, in our case diabetics and nondiabetics, and for the full dataset. We use `nototals` to suppress those medians for the full dataset.

```
. use https://www.stata-press.com/data/r18/nhanes2l
(Second National Health and Nutrition Examination Survey)
. table (var) (diabetes),
> statistic(median age weight bpsystol tcresult iron) nototals
```

	Diabetes status	
	Not diabetic	Diabetic
Age (years)	48	64
Weight (kg)	70.19	74.84
Systolic blood pressure	128	142
Serum cholesterol (mg/dL)	212	223
Serum iron (mcg/dL)	96	88

We would also like to perform a rank-sum test for each of those variables to test whether the distributions are the same across the categories of `diabetes`. If we wanted to perform the test only for `age`, we could type

```
. ranksum age, by(diabetes)
```

Because we want to perform the test for multiple variables, we write a loop to issue the `ranksum` command for each variable. We use the `collect` prefix to collect the two-sided p-value (`r(p)`). The `tag()` option tags the results with the dimension `var`, which will allow us to align these results with the medians we computed above.

```
. foreach x in age weight bpsystol tcresult iron {
  2.         quietly: collect r(p), tag(var['x']): ranksum 'x', by(diabetes)
  3. }
```

We want to create a table with the medians we computed with `table` and the p-values we collected with the `collect` prefix. `collect` stored the results in the current collection, so we have the results all in one place. Now, we can use `collect layout` to arrange the items from the collection into a table. Again, we place the variables on the rows and the levels of `diabetes` and the statistics from `ranksum` on the columns.

```
. collect layout (var) (diabetes result)

Collection: Table
      Rows: var
   Columns: diabetes result
   Table 1: 5 x 3
```

(*output omitted*)

We omit the table preview here because of the table's width.

Customizing the table

The table above is wide because of the long label for the p-values. We can see the labels by using the `collect label list` command with the `result` dimension.

```
. collect label list result

  Collection: Table
   Dimension: result
       Label: Result
Level labels:
           N  Sample size
         N_1  Sample size of first group
         N_2  Sample size of second group
       Var_a  Adjusted variance
      group1  Value of variable for first group
      median  Median
           p  Two-sided p-value from normal approximation
         p_l  Lower one-sided p-value from normal approximation
         p_u  Upper one-sided p-value from normal approximation
     sum_exp  Expected sum of ranks for first group
     sum_obs  Observed sum of ranks for first group
           z  Z statistic
```

The p-values correspond to the level `p` of the dimension `result`. Below, we modify this label with `collect label levels`. Then, we preview our table:

```
. collect label levels result p "p-value", modify
. collect preview
```

	Diabetes status		p-value
	Not diabetic	Diabetic	
Age (years)	48	64	9.33e-69
Weight (kg)	70.19	74.84	1.12e-10
Systolic blood pressure	128	142	3.61e-43
Serum cholesterol (mg/dL)	212	223	.0000178
Serum iron (mcg/dL)	96	88	2.17e-08

Because labels for the levels of `diabetes` are descriptive enough, we can hide the title for the dimension. We format the p-values to have three decimal places. We also remove the vertical border. Then, we preview our table once more:

```
. collect style header diabetes, title(hide)
. collect style cell result[p], nformat(%5.3f)
. collect style cell border_block, border(right, pattern(nil))
. collect preview
```

	Not diabetic	Diabetic	p-value
Age (years)	48	64	0.000
Weight (kg)	70.19	74.84	0.000
Systolic blood pressure	128	142	0.000
Serum cholesterol (mg/dL)	212	223	0.000
Serum iron (mcg/dL)	96	88	0.000

In fact, we prefer not to report our p-values in this form. Instead, we can display them as being less than 0.001. With `collect style cell`, we can specify a minimum value, and any p-values smaller than the minimum will be displayed as simply less than that minimum:

```
. collect style cell result[p], minimum(0.001)
. collect preview
```

	Not diabetic	Diabetic	p-value
Age (years)	48	64	<0.001
Weight (kg)	70.19	74.84	<0.001
Systolic blood pressure	128	142	<0.001
Serum cholesterol (mg/dL)	212	223	<0.001
Serum iron (mcg/dL)	96	88	<0.001

See [TABLES] **collect style header** and [TABLES] **collect style cell** for more information on the commands we used here to customize the table.

Reference

McDowell, A., A. Engel, J. T. Massey, and K. Maurer. 1981. Plan and operation of the Second National Health and Nutrition Examination Survey, 1976–1980. *Vital and Health Statistics* 1(15): 1–144.

Also see

[R] **table** — Table of frequencies, summaries, and command results

[TABLES] **collect get** — Collect results from a Stata command