# Basic Data Analysis Course on Stata

Md Monowar hossain

December 2, 2024

# About Stata software

- Stata is a software package used for statistical analysis. The name Stata is a syllabic abbreviation of the words statistics and data.
- Stata was developed by StataCorp in 1985.
- Stata has always emphasized a command-line interface, which facilitates replicable analyses. Starting with version 8.0, however, Stata has included a graphical user interface which uses menus and dialog boxes.
- The latest release is Stata version 17, released in April, 2021.

# Overview of Stata file extensions

- Stata data file: .dta, example: mydata.dta
- Stata syntax file: .do, example: mysyntax.do
- Stata output file: .smcl, example: myoutput.smcl, smcl stands for
- Stata Markup and Control Language

# Stata windows

There is an additional window for graph that appears when a graph is produced in Stata.

- Command window: where commands are entered. Press the Enter key on your keyboard to execute any command.
- Results window: where results appear (i.e, are printed).
- Variables window: lists the variables in the current dataset.
- Review window: lists past (i.e, already executed) commands.
- Graph window: where graphs are displayed (appears when graphs are generated).

## GUI and CLI

- You can either use the menu-based **Graphical User Interface** (GUI) or the command-driven **Command Line Interface** (CLI) to interact with Stata. But there are some advantages of using CLI:
  - **Command syntax** has the advantage of reproducibility.
  - Many of the commands are short and easy to execute, and as proficiency grows, it is often faster to type commands rather than use the menu.

- We will mostly cover command syntax in this course. We will be providing commands so that you will become familiar with them. The commands will be useful for constructing longer and more complicated programs.
- The command executes if it is entered properly
- Issues an error message (in red) if it cannot figure out what to do, or if there is something which could inadvertently change the dataset.
- If Stata gives you an error, it will provide you with a return code, r(#). You can find out more about the error by typing: lookup rc #.
- To get help regarding any command, e.g. the regression command, type: help regression

# Rules for variable naming in Stata

- Stata is case-sensitive, so using all lower case letters in variable names is a good idea.
- They can contain no more than 32 characters.
- They can contain letters, numbers, or underscores ( ).
- Spaces or other special characters (like &, *, %, etc.) are not allowed.
- The first character must be a letter or underscore, not a number.
- Starting variable names with underscores is a really bad idea, since Stata's built-in variables begin with an underscore.

# In the morning

### Getting up

- Turn off alarm
- Get out of bed

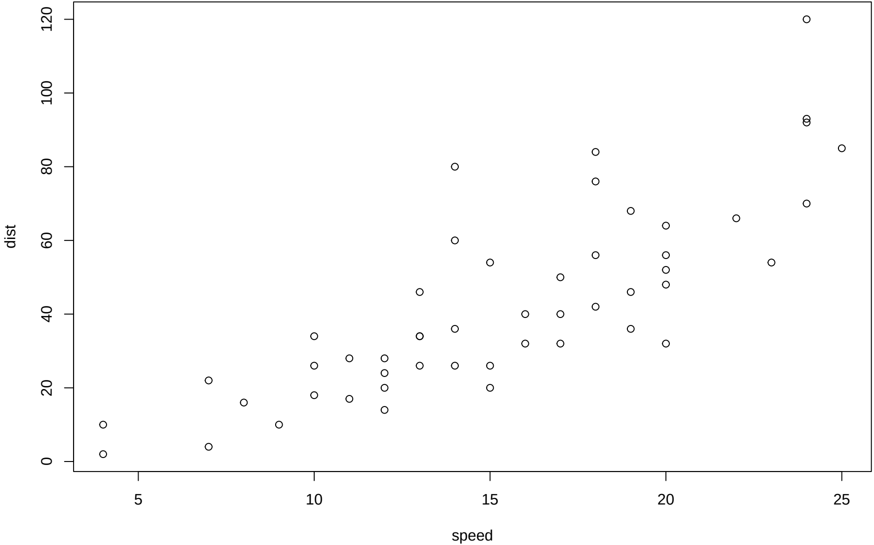
### Breakfast

- Eat eggs
- Drink coffee

# In the evening

### Dinner

- Eat spaghetti
- Drink wine


0
20
40
60
80
100
120
dist
5
10
15
20
25
speed

## Stata windows

There is an additional window for graph that appears when a graph is produced in Stata.

- Command window: where commands are entered. Press the Enter key on your keyboard to execute any command.
- Results window: where results appear (i.e, are printed).
- Variables window: lists the variables in the current dataset.
- Review window: lists past (i.e, already executed) commands.
- Graph window: where graphs are displayed (appears when graphs are generated).

### GUI and CLI

- You can either use the menu based Graphical User Interface (GUI) or the command-driven Command Line Interface (CLI) to interact with Stata. But there are some advantages of using CLI.
- Command syntax also has the advantage of reproducibility.
- Many of the commands are short and easy to execute, and as proficiency grows; it is often faster to type commands rather than use the menu.

## GUI and CLI

We will mostly cover command syntax in this course. We will be providing commands so that you will become familiar with them. The commands will be useful for constructing longer and more complicated programs.

- The command executes if it is entered properly Issues an error message (in red) if it cannot figure out what to do, or if there is something which could inadvertently change the dataset.
- If Stata gives you an error, it will provide you with a return code, r(#). You can find out more about the error by typing: lookup rc

1.

- To get help regarding any command, e.g. the regression command, type: help regression

### Rules for variable naming in Stata

- Stata is case-sensitive, so using all lower case letters in variable names is a good idea.
- They can contain no more than 32 characters.
- They can contain letters, numbers, or underscores ( ).
- Spaces or other special characters (like &, *, %, etc.) are not allowed.
- The first character must be a letter or underscore, not a number. Starting variable names with underscores is a really bad idea, since Stata's built-in variables begin with an underscore.

## Data editor

- You are allowed to enter, view, or edit your data in “data editor”. It looks like a spreadsheet.
- Typically, variables are listed across the top, and cases are listed down the side.
- To just have a view of your data, you need to type the command browse in command windows; in the browse mode you cannot change your data.
- To enter or edit your data you need to type edit which will take you to the edit mode.

## Do file

- A do file contains a list of Stata commands.
- The do file can be created using Stata's do file editor.

## Directory management

- To check the present working directory:

```
pwd
```

- To change the current working directory:

```
cd "E:\"
```

- To make a new directory inside the current working directory:

```
mkdir "E:\StataClass1"
```

- To make the newly created directory as working directory:

```
cd "E:\StataClass1"
```

## Log file

- Log files allow you to record the inputs and outputs from Stata.Output sent to the results screen is not automatically saved. So you should always set up a log file to record your work.
- By default, Stata produces log files in its own format (smcl–Stata Markup and Control Language).

## log files

- To start a log file, type: log using example.txt, replace
- Here replace tells Stata to overwrite any existing version of example.txt.
- We can also type: log using example.txt, append
- Here append adds new input/output to end of existing log file example.txt.
- A good approach to permanently storing log files with important results is to rename them after the analysis is complete - e.g. example class1.txt.
- Type log off to pause recording.
- Type log on to resume recording.
- Type log close to end the log file.

## Entering data into Stata

- You can enter data into Stata using GUI by clicking the Edit button or by typing edit in the command window.
- Also, we can use a do file to enter data by writing the following syntaxes:

```
input hhid sex str10 location
10030 1 urban
10031 0 rural
10032 1 rural
end
```

where hhid and sex are numeric variables and location is string variable. - For any string variable we need to specify the maximum number of characters the string will have using str#, where # indicates that location can have maximum characters 10.

## Value labels and the color coded system

In the example used above we have 1 and 0 to represent sex. Say, 0=Male and 1=Female. If we want to label the values by 'Male' and 'Female', then first we have to define a value label. This label will be created as an object in Stata. Then we will need to attach that value label to a variable. For example,

```
label define gender 0 "Male" 1 "Female"
label values sex gender
```

Here we create a value label gender as an object in the first step. Then we attach it to the variable sex. There is a color coded system in Stata. All the numeric variables appear in black. The string variables appear in red. The labelled variables appear in blue.

## Labelling variables and saving data

Labeling a variable means giving the variable names a definition. For example, we mean ‘household identification number’ by hhid and ‘type of area of residence’ by location. It is always advisable to label the variables for future use. We can do that as follows:

```
label variable hhid "Household identification number"
label variable location "Type of area of residence"
label variable sex "Gender of the respondent"
```

After we are done with the initial formatting, we can save the data we have just entered and labeled. For example,

```
save "E:\StataClass1\FirstData.dta"
```

## Converting data into different types

- String to labeled numeric:

```
encode location, gen(nlocation)
labelbook nlocation
```

- Labeled numeric to string:

```
decode sex, gen(nsex)
```

## Entering data using Data Editor

- Now enter the previous data.

## Reading a Stata data file

- Reading a data file from a particular location:

```
use "E:\bd_individual.dta", clear
```

- Reading a data file from a working directory:

```
use bd_individual.dta, clear
```

## Reading a Stata data file from a website

```
use http://www.stata-press.com/data/r14/systolic, clear

webuse query

webuse lifeexp

webuse apple

webuse set http://www.zzz.edu/users/~sue
```

## Exporting data files

- Often, we need to save the entered dataset as .csv or .txt format, so that it can be read in different software.
- To create such a .csv file, we have to export our data as follows:

```
outsheet using "E:\bd_individ.csv",comma nolabel replace
```

- Here, comma is the delimiter, nolabel means we want the values of the labeled variables as outputs, not the labels themselves, replace replaces any previous file in the same name.
- To export the data into .txt format we can use:

```
  outsheet using "E:\bd_individ.txt", nolabel replace
```

## Exporting data files

- If we want it to save the data in excel format, then we can write the following syntaxes: export excel using bd_individ.xls, /// firstrow(variables) nolabel replace
- Here, firstrow(variables) indicates we want to save the variable names into the first row of the excel file.

### Importing data files

- We can read dataset in .csv or .txt format, by using the following commands:

insheet using bd_individ.csv, clear insheet using bd_individ.txt, clear - To read an excel file we will use the following command:
import excel bd_individ.xls, /// sheet(“Sheet1”) firstrow clear - Here, firstrow means we can use variable names contained in the first row of the excel file as the variable names of the Stata file.

### Renaming a variable

- rename Sometimes you have to rename the variables so that the name can give you an idea of what the variable is about. For example,

use bd_individual.dta, clear rename v011 DOB rename v012 current_age

## Labeling a data set

- label data

A data label is a description of the entire dataset. Dataset labels are displayed when you use describe command. It helps remind you what the dataset was about. For example,
label data /// “Data on Bangladeshi females at reproductive ages”

- labelbook This command provides with the details about a value label, such as ranges of values and some other basic information about labels. For example,

labelbook V024 - Note that V024 is a value label, not the variable v024.

### Exploring/examining your dataset

It is a good idea to examine your data when you first read it into Stata.

- There may be unexpected data
- There may be missing data
- There may be miscoded data

- browse or edit

The command browse brings up a spreadsheet-style data editor in which you can examine the raw data. But often we may only want to view a few variables at a time, especially in large datasets with a large number of variables. In such cases, simply list the variables you want to examine after browse. For example,

browse caseid v002 v003 v012 v024

- We can also browse data by putting a condition. For example,

browse if v012<21 browse if v024==1 browse caseid v002 v003 v012 v024 if v012<21

- We can use edit to correct if any anomaly is found during the examination of data.

- list The browse and edit commands start a pop-up window in which you can examine the raw data. You can also examine it within the results window using the list command-although listing the entire dataset is only feasible if it is small. If the dataset is large, you can use some options to make the output of list more tractable. For example,

list list caseid v002 v003 v012 v024 list if v024==1 list caseid v002 v003 v012 v024 if v024==1

- assert

With large datasets, it is often impossible to check every single observation using list or browse. An useful command is assert which verifies whether a certain statement is true or false. For example, you might want to check whether ages of the respondents in our dataset is between 15 and 49 as they should be.

assert v012>=15 & v012<=49 (blank output means no contradiction found) assert v024>15 & v024<=49 17863 contradictions in 17863 observations assertion is false

- If the statement is true, assert does not yield any output on the screen. If it is false, assert gives an error message and the number of contradictions.

- describe

This command reports some basic information about the dataset and its variables (e.g. size, number of variables and observations, storage types of variables, labels etc.). You can also mention some specific variables to see information on only those. For example,

describe describe caseid v002 v003 v012 v024 - summarize This command provides summary statistics, such as means, standard deviations, and so on. For example,

summarize v012 summarize v012, detail /*(to get percentiles)*/ summarize v012 if v024==1

- codebook This command provides various information on variables such as range of values, numbers of observations, missing values and unique cases, summary statistics of numerics etc.

codebook codebook caseid v002 v003 v012 v024

- tabulate This command provides one-way or two-way frequency distribution for categorical variables. For example,

tabulate v024 tabulate v024 v106

## Creating new variables

- Sometimes we may need to create new variables those are function of some other variables.
- We can easily do that in Stata using function gen or egen.
- The former one is the basic command and the later one is an extension.
- The gen function is used to create a new variable which is a function of one or more other variables.
- The egen function has special functions such as mean, median, total, etc. that can be used to create a new variables.

## Essential operators

- We need to know some arithmetic, logical and comparison operators in order to subset the data or combine variables in Stata.
- The operators of Stata are listed below: Arithmetic Logical Comparison +————————— —————————- —————————-
- addition ~ 0r ! not > greater than
- subtraction | or < less than
- multiplication & and >= > or equal / division <= < or equal ˆ power == equal ~= not equal

————————————————————————————

- gen For example, we want to calculate the age difference between husband and wife for the BDHS data using variables v730 and v012. This can easily be carried out as,

gen aggap = v730 - v012 - egen Now, if we want a variable whose values are obtained by the mean age of women who belong to the same education group.

egen nage=mean(v012), by(v106) - gen ecb=0 replace ecb=1 if v212<18

## Recoding variable values

- Values of a variable can be recoded using function recode
- recode changes the values of numeric variable according to the rules specified
- Values that do not meet any of the conditions of the rules are left unchanged, unless an otherwise rule is specified.
- Basic form of the syntax is recode varlist (rule) (rule) . . . , generate(newvar)
- The most common forms for rule are
  +——————————————————————-+ | Rule | Example | Meaning | |————–+————–+——————————| | # = # | 3 = 1 | 3 recoded to 1 | | # # = # | 2 . = 9 | 2 and . recoded to 9 | | #/# = # | 1/5 = 4 | 1 through 5 recoded to 4 | | nonmissing = # | nonmissing = 8 | all other nonmissing to 8 | | missing = # | missing = 9 | all other missings to 9 | +——————————————————————-+

- This command changes the values of numeric variables according to the rules specified. For example,

recode v106 (2 3 = 2), generate(nv106) label define newlab 0 “no education” 1 “primary” /// 2 “secondary or higher” label values nv106 newlab - You can also categorize a numeric variable by using recode command and create a new variable using generate with a new value label defined within label. For example,

summarize v012 /*respondent’s current age*/ recode v012 (15/17 = 0 “<18”) (18 19 = 1 “18-19”) /// (20/29 = 2 “20-29”) (30/39 = 3 “30-39”) /// (else=.), generate(agegroups) label(agegrp)

## Sub-setting data

keep
You can keep a subset of observations or variables for analysis and delete the others using this command. For example,
keep caseid v002 v003 v012 v024 keep if v012>=18 To keep the first five observations: keep in 1/5
drop You can subset the data by dropping some observations or variables using this command. For example, drop v730 drop if v012<18

### Sorting data

sort This command arranges the dataset according to ascending order of a variable. You can also specify more than one variable to sort.
sort caseid sort v190 v024 /*division sorted within wealth index*/ browse caseid v190 v024

## Sorting data

gsort This command stands for generalized sorting. This can arrange data both in ascending and descending order. You can just put a plus or minus sign before the variable to indicate ascending and descending order respectively. For example,
gsort +v190 /*or simply: gsort v190*/ gsort -v190 gsort v190 -v024 browse caseid v190 v024

## Reshaping datasets

- Sometimes it is needed to reshape data such as from wide to long and vice versa.
- How do you know if your data is in wide or long format?
- If there is only one unique case ID per row, then your data is in wide format.
- If there are several rows with the same case ID, then your data is in long format.
- Stata can do that simply using reshape command
- Now to convert the dataset from wide to long format we write: use faminc.dta, clear reshape long faminc, i(famid) j(year)
- To convert the dataset from long to wide format we write: use kids.dta, clear reshape wide age, i(famid) j(birth)

## Appending datasets

- You can combine different datasets into a single large dataset using the append and merge commands.
- First we create two separate data files by sub-setting based on age of respondents to obtain women aged less and greater than 18 years.

use bd_individual.dta, clear keep if v012>=18 save adults.dta count
use bd_individual.dta, clear keep if v012<18 save nonadults.dta count

### Appending datasets

- We can now use append to join these two datasets. The command append is used to add extra observations (rows) from another dataset to the existing dataset. append using adults.dta count
- Note that, you have to have the same number of variables in both data files before you append them.

## Merging datasets

- Now we create two datasets by sub-setting based on demographic and wealth information of the respondents. use bd_individual.dta, clear keep caseid v001 v002 v003 v010 v024 v025 v106 sort caseid save demographic.dta

use bd_individual.dta, clear keep caseid v119 v120 v121 v122 v123 v124 v125 sort caseid save wealth.dta - We have to sort the two datasets based on a common variable beforehand in order to merge them.

## Merging datasets

- We can now employ merge to join these two datasets. merge 1:1 caseid using demographic.dta
- Note that, the last column produced indicates the case was present in both master and secondary data.
- The open dataset is called the master data.
- For cases that come from only master data will show 1 in the column produced. For cases from only secondary data it will show 2.

## Merging datasets

- Now, we perform 1:m merge as follows: use data1.dta, clear merge 1:m household_id using data2.dta
- We can also perform m:1 merge using the following codes: use data2.dta, clear merge m:1 household_id using data1.dta

## Available graphs

- For qualitative data
- Bar chart
- Pie chart
- For quantitative data
- Histogram
- Boxplot
- Scatter diagram

### Graphical presentation of qualitative data

- Bar chart and pie chart are graphical presentation for summary of qualitative data.
- In bar chart comparisons are made based on original scale of measurement, while in pie chart comparisons are made in terms of transformed measurement scale (in degree)
- Therefore, bar chart is more preferable than pie chart.

## Bar chart

- A bar graph is another tool to represent frequencies or percentages of categories of a nominal or ordinal variable.
- In a bar graph, categories are displayed through rectangles of equal width where heights of the rectangles represent frequency/percentage of the respective categories.

### Types of bar chart

The bar charts can be classified as the following way

- Simple bar chart (involves with single variable)
- Multiple bar chart (involves with multiple variables)
- Component (or Stacked) bar chart (involves with multiple variables)

### Simple bar chart

use birthwt.dta, clear
graph bar (count), over(low) /// ytitle(“Frequency”) /// title(“Frequency distribution of low”) /// note(“Source: Baystate Medical Centre, Springfield, MA”)
graph bar, over(low) /*gives a percentage bar graph* /
How to save these graphs?

### Multiple and stacked bar charts

graph bar (count), over(race) over(low) graph bar, over(race) over(low) graph bar, over(race) over(low) stack asyvars /*treat first over() group as yvars*/

### Pie chart

- The pie diagram is intended to compare the distinct components which together constitute a whole.
- The whole is presented by a circle of arbitrary radius and segments of the circle represents the component parts.
- The whole corresponds to the 360 ▫ which is the total number of degree in a circle. This 360 ▫ is to be divided proportionately among the components for drawing a pie chart.

### Pie chart

```
graph pie, over(low)
```

## Graphical presentation of quantitative data

- A quantitative variable can be presented using different graphs among which histogram and boxplot are very popular.
- The former requires construction of frequency table and the later does not.
- A histogram looks like bar diagram, but it differs by the fact that rectangles touch one another and can be of different widths.
- However, in case of histogram, bar height can represent either the frequency or density.

### Histogram

```
hist bwt, frequency /// title("Histogram for birth weight") ///
subtitle("(in gram)") /// xtitle("Birth weight") /// ytitle("Frequency")
```

## Histogram

```
hist bwt, fraction
hist bwt, bin(8) fraction normal
hist bwt, width(1000) fraction normal
hist bwt, width(1000) start(500) fraction normal
```

### Boxplot

```
graph box bwt
graph hbox bwt
graph box bwt, over(smoke)
```

## Scatter plot

```
twoway scatter bwt age
```

## Line diagram

- Line diagrams are used to track changes over different periods of time.
- For example, assume that on day one of trading, a given stock's price was $30, resulting in a data point at (1, $30). On day two of trading, the stock's price was $35, resulting in a data point at (2, $35).
- Each data point can be plotted and connected by a line that visually would show the changes in the values over time.
- If the value of the stock increased daily, the line would slope upward. Conversely, if the price of the stock was steadily decreasing, then the line would slope downward.

### Line diagram

use https://www.stata-press.com/data/r17/uslifeexp /*(U.S. life expectancy, 1900-1999)*/ line le year line le le_male le_female year

## About the data

- We are going to use the national-scale dataset wm.dta from Bangladesh Multiple Indicator Cluster Survey (MICS) 2019.
- The data were collected by the Bangladesh Bureau of Statistics in cooperation with the UNICEF Bangladesh, as part of the global MICS programme.
- The survey employed a two-stage stratified cluster sampling approach where the 64 districts were the sampling strata.
- The 2011 national census enumeration areas (EAs) were defined as the primary sampling units (clusters) and households as the secondary sampling units.
- In the first stage of sampling, 3220 EAs were selected with probability proportional to size (PPS) method.
- In the second stage, a systematic sample of 20 households was obtained from each of the selected EAs, which led to a final sample of 64,400 households for the survey.

### One-way table for qualitative data

```
use wm.dta, clear
tab HH7
tab HH7 [iweight=wmweight] /* iweight means ‘importance weight’ */
```

## One-way table for quantitative data

```
recode WB4 (15/20 = 1 “15-20”) (20/25 = 2 “20-25”) /// (25/30 = 3
“25-30”) (30/35 = 4 “30-35”) /// (35/40=5 “35-40”) (40/45=6
“40-45”) /// (45/50=7 “45-50”), generate(agegroups) label(agegrp)
tab agegroups [iweight=wmweight]
```

### Two-way table for qualitative data

```
tab HH6 welevel [iweight=wmweight]
tab HH6 welevel [iweight=wmweight], row
tab HH6 welevel [iweight=wmweight], col
tab HH6 welevel [iweight=wmweight], cell
```

### Descriptive statistics

```
summarize WB4 summarize WB4, detail
summarize WB4 [aweight=wmweight]
summarize WB4 [aweight=wmweight], detail
```

### Descriptive statistics

```
by HH6, sort: summarize WB4, detail
by HH6, sort: summarize WB4 [aweight=wmweight], detail
codebook HH6 codebook HH7
keep if HH6==1 summarize WB4
keep if HH6==1 & HH7==10 summarize WB4
```

## Declare survey design for dataset: svyset

svyset manages the survey analysis settings of a dataset. You use svyset to designate variables that contain information about the survey design, such as the sampling units and weights.
svyset [pw=wmweight], psu(WM1) strata(HH7A) /* WM1 is the cluster number HH7A is the district */
tab HH7 [iweight=wmweight] svy: tab HH7 /* both the commands give the same result */
svy: tab HH6 welevel, row pearson /* pearson chi-square */

## Multiple response analysis

```
use multipleresponse.dta, clear
ssc install mrtab
mrtab OTHER2_1 OTHER2_2 OTHER2_3 OTHER2_4 OTHER2_5 ///
OTHER2_6 OTHER2_7 OTHER2_8 OTHER2_9 OTHER2_10 ///
OTHER2_11 OTHER2_12 OTHER2_13 OTHER2_14 OTHER2_15 ///
OTHER2_16 OTHER2_17 OTHER2_18 OTHER2_19 OTHER2_20 ///
OTHER2_21 OTHER2_22 OTHER2_23 OTHER2_24 OTHER2_25 ///
OTHER2_26 OTHER2_27
```

### Exporting results

- Copy table to a excel file
- Copy table to a word file
- Copy figure to a word file
- Create a pdf file of output

### One sample t-test

- A one sample t-test allows us to test whether a population mean (of a normally distributed interval variable) significantly differs from a hypothesized value.
- For example, using the birthwt.dta data file, say we wish to test whether the average birth weight of newborns differs significantly from 2500 gm.

## One sample t-test

use birthwt.dta, clear
ttest bwt=2500
One-sample t test
+———————————————————————————— Variable |
Obs Mean Std. Err. Std. Dev. [95% Conf. Interval]
———+————————————————————————— bwt | 189
2944.587 53.04254 729.2143 2839.952 3049.222
+———————————————————————————— mean =
mean(bwt) t = 8.3817 Ho: mean = 2500 degrees of freedom = 188 Ha: mean < 2500 Ha: mean != 2500 Ha: mean > 2500 Pr(T < t) = 1.0000 Pr(|T| > |t|) = 0.0000 Pr(T > t) = 0.0000 - The mean of the variable bwt for this particular sample is 2944.587, which is significantly different from the test value of 2500. - We would conclude that the mean birth weight is significantly higher than 2500.

### Two independent samples t-test

- An independent samples t-test is used when you want to compare the means of a normally distributed interval dependent variable for two independent groups.
- For example, we may wish to test whether the mean birth weight is the same with smokers and nonsmokers.
- The test variable is bwt and group variable is smoke.

## Two independent samples t-test

```
ttest bwt,by(smoke)
Two-sample t test with equal variances
+———————————————————————————————————— Group |
Obs Mean Std. Err. Std. Dev. [95% Conf. Interval]
————+———————————————————————————————— no | 115
3055.696 70.18559 752.6566 2916.659 3194.733 yes | 74 2771.919 76.681
659.6349 2619.094 2924.744
————+———————————————————————————————— combined |
189 2944.587 53.04254 729.2143 2839.952 3049.222
————+———————————————————————————————— diff |
283.7767 106.9688 72.75612 494.7973
+———————————————————————————————————— diff =
mean(no) - mean(yes) t = 2.6529 Ho: diff = 0 degrees of freedom = 187
Ha: diff < 0 Ha: diff != 0 Ha: diff > 0 Pr(T < t) = 0.9957 Pr(|T| > |t|)
= 0.0087 Pr(T > t) = 0.0043
```

- The results indicate that there is a significant difference between the means with smokers and nonsmokers, since the p-value is 0.0087. - More specifically, the mean birth weight of

## One-way ANOVA

- A one-way analysis of variance (ANOVA) is used when you have a categorical independent variable (with two or more categories) and a normally distributed interval dependent variable and you wish to test for differences in the means of the dependent variable broken down by the levels of the independent variable.
- For example, we may wish to test whether the mean birth weight differs among the three races.

## One-way ANOVA

oneway bwt race

Analysis of Variance Source SS df MS F Prob > F
+———————————————————————— Between
groups 5015725.25 2 2507862.63 4.91 0.0083 Within groups 94953930.6 186 510505.003
+———————————————————————— Total
99969655.8 188 531753.488 Bartlett's test for equal variances: chi2(2) = 0.6595 Prob>chi2 = 0.719 - As indicated in the ANOVA table, the mean of birth weight differs significantly among the levels of races. That is, the means of birth weight across the races are not the same.

## Paired t-test

- A paired (samples) t-test is used when you have two related observations (i.e. two observations per subject) and you want to see if the means on these two normally distributed interval variables differ from one another.
- Assume that twenty subjects participated in an experiment to study the effectiveness of a certain diet, combined with a program of exercise, in reducing serum cholesterol levels. Data are recorded on the serum cholesterol levels for the 10 subjects at the beginning of the program (before) and at the end of the program (after), and is available in the data file diet.
- The question to be answered is: Do the data provide sufficient evidence for us to conclude that the diet-exercise program is effective in reducing serum cholesterol levels?

## Paired t-test

```
use diet.dta, clear
ttest after=before
Paired t test
+——————————————————————————————— Variable |
Obs Mean Std. Err. Std. Dev. [95% Conf. Interval]
———+—————————————————————————————— after | 10
226.8 9.219303 29.154 205.9445 247.6555 before | 10 244.6 10.77775
34.08225 220.219 268.981
———+—————————————————————————————— diff | 10
-17.8 4.762819 15.06136 -28.57425 -7.025755
+———————————————————————————————
mean(diff) = mean(after - before) t = -3.7373 Ho: mean(diff) = 0
degrees of freedom = 9 Ha: mean(diff) < 0 Ha: mean(diff) != 0 Ha:
mean(diff) > 0 Pr(T < t) = 0.0023 Pr(|T| > |t|) = 0.0046 Pr(T > t) =
0.9977 - These results indicate that the diet-exercise program is
significantly effective in reducing serum cholesterol levels.
```

## Test for single proportion

- A one sample proportion test allows us to test whether the proportion of successes on a two-level categorical dependent variable significantly differs from a hypothesized value.
- For example, we may wish to test whether the proportion of mothers with low birth weight babies differs signiffcantly from 0.40.
- The prtest command assumes that the variables it will act on are binary (0/1) variables and the proportion of interest is the proportion of 1'

## Test for single proportion

```
prtest low=0.4
One-sample test of proportion low: Number of obs = 189
+------------------------------------------------------------ Variable |
Mean Std. Err. [95% Conf. Interval]
---------+-------------------------------------------------- low |
.3121693 .0337058 .2461071 .3782315
+------------------------------------------------------------ p =
proportion(low) z = -2.4647 Ho: p = 0.4 Ha: p < 0.4 Ha: p != 0.4 Ha: p
> 0.4 Pr(Z < z) = 0.0069 Pr(|Z| > |z|) = 0.0137 Pr(Z > z) = 0.9931
```

- The results indicate that the the proportion of mothers with low birth weight babies is significantly lower than the hypothesized value of 40%.

## Test for two proportions

```
prtest ht=smoke
Two-sample test of proportions ht: Number of obs = 189 smoke: Number
of obs = 189
+—————————————————————————————— Variable |
Mean Std. Err. z P>|z| [95% Conf. Interval]
————-+————————————————————————- ht |
.0634921 .0177372 .0287278 .0982563 smoke | .3915344 .0355036
.3219487 .4611201
————-+————————————————————————- diff |
-.3280423 .0396877 -.4058287 -.2502559 | under Ho: .0431254 -7.61
0.000 +—————————————————————————— diff
= prop(ht) - prop(smoke) z = -7.6067 Ho: diff = 0 Ha: diff < 0 Ha: diff
!= 0 Ha: diff > 0 Pr(Z < z) = 0.0000 Pr(|Z| > |z|) = 0.0000 Pr(Z > z)
= 1.0000
```

- The results indicate that the the proportion of mothers with hypertension is significantlylower than the proportion of mothers who are smokers

### Equivalence of two proportions test and the chi-square test of independence

tab low smoke, chi2 /* test of independence */
prtest smoke, by(low) /* compares between the proportion of smokers(in low-yes) and proportion of smokers (in low-no)*/

- If you square the z value then you will obtain the chi-square value.
- p-values for both the tests are the same.

### Chi-square goodness of fit test

- A chi-square goodness of fit test allows us to test whether the observed proportions for a categorical variable differ from hypothesized proportions.
- For example, suppose we believe that the women under different races are of equal proportions.
- We want to test whether the observed proportions from our sample differ significantly from the hypothesized equality of proportions.

## Chi-square goodness of fit test

```
findit csgof
csgof race


+--------------------------------------+
| race expperc expfreq obsfreq |
|--------------------------------------|
| white 33.33333 63 96 |
| black 33.33333 63 26 |
| other 33.33333 63 67 |
+--------------------------------------+
chisq(2) is 39.27, p = 0
```

- Since the p-value is very small, we can reject the null hypothesis that proportion of women is the same for all the races.

## Chi-square goodness of fit test

```
csgof race, expperc(33.3,33.3,33.3)

+-----------------------------------+
| race expperc expfreq obsfreq |
|-----------------------------------|
| white 33.3 62.937 96 |
| black 33.3 62.937 26 |
| other 33.3 62.937 67 |
+-----------------------------------+

chisq(2) is 39.31, p = 0
```

## Chi-square goodness of fit test

- Suppose we want to test whether the observed proportions in the sample differ significantly from a set of hypothesized proportions.

```
csgof race, expperc(50,20,30)


+-------------------------------------+
| race expperc expfreq obsfreq |
|-------------------------------------|
| white 50 94.5 96 |
| black 20 37.8 26 |
| other 30 56.7 67 |
+-------------------------------------+


chisq(2) is 5.58, p = .0615
```

- Note that we cannot reject the null hypothesis at 5% level of significance