L5. Cleaning data

Before you begin analyzing any data, it is crucial to understand it, clean it and recode all your variables of interest properly. We can think of 5 big steps to starting a project and cleaning data:

1. Inspect your variables of interest (inspect, codebook, browse, etc.)
   * Understand the data structure (long – wide).
   * Understand how variables are measured. The official codebook may not be enough.
2. Look at descriptive statistics (frequencies, means, standard deviations, distribution, etc.)
   * Make sure your data looks reasonable and there are no potentially influential outliers! (e.g. crazy years of birth, negative number of children)
   * Check for unusual values (e.g., those people who had sex 88 times a week)
3. Missing data: do we have a lot? Is it worth imputing? (we’ll talk later about it)
   * Make sure missingness is coded correctly. In Stata, recode to “.” (or “.letter”)!
4. Check for potential transformations.
   * After looking at the distribution, it may be the case that the form of the variable is not optimal for regression.
5. Recodes:
   * After understanding the nature of our variables of interest, we go on ahead and recode things appropriately.

**Missing data:**

Missing values are values in a data set values that were not recorded within the variable’s value range. Missing values can have many causes including a respondent's refusal to answer survey questions, an interviewer incorrectly coding a response, or questions that do not apply to a respondent.

The basic missing value for numeric variables is represented by a dot ( . ). Stata calls these *soft missing*. On the other hand, we can also separate types of missing data by including a letter after the dot from .a to .z. Stata calls these *hard missing*. Usually, hard missing tags are used to identify values we know about. For example: Survey data often use codes such as 88 for not applicable and 99 for not ascertained. For example age at marriage may be coded 88 for single women and 99 for women who are known to be married but did not report their age at marriage. You need to be careful with these variables, because these cases will influence your estimates due to how high they are. You need to let Stata know they are missing values. You will often want to distinguish these two cases using different kinds of missing value codes. If you wanted to recode 88's to .n (for "na" or not applicable) and 99's to .m (for "missing") you could use the code.

A very useful way of checking out your missing data on your variable is using the descriptive commands we say earlier. Personally, I prefer to use the tabulate function with the options of missing and nolab.

**Missing data and regression. What is complete case analysis?**

Until we learn how to impute data, we will use complete case analysis. As the name indicates, this type of analysis only uses “complete” cases, or cases where the data are fully observed. The name “listwise deletion” is more intuitive for some people because complete case analysis is accomplished by removing from the analysis any cases that have missing data on any variables in the model. No matter how you say it, it means that any case that has missing data will be excluded from analysis. Here is an example:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id variable | Income | Gender | Age | Years of Education |
| 1 | $$$$ | M | 46 | 12 |
| 2 | . | F | 23 | 14 |
| 3 | $$$$$$ | M | 67 | 16 |
| 4 | $$$$ | F | 19 | . |
| 5 | $$$$ | F | 26 | 18 |
| 6 | $$$ | . | 34 | 20 |
| 7 | . | M | 44 | 11 |
| 8 | $$$$$$$ | F | 21 | 24 |

Since we can only keep those cases that have complete information, we will drop those that have missing data in any variable of our analysis (in red). Further down the line we will have to pay attention to whether there is a given pattern in the missingness or it is randomly distributed. Additionally, we really want to pay attention to whether the total amount of missing data using listwise deletion is high. Usually, the cut-off point is 10%. If your amount of missingness exceeds 10% you may want to consider multiple imputation -at the very least, as a robustness check. In the example above, our amount of missing would be of 50% since we have to discard 4 cases and our sample is of 8 people.

**Data transformations:**

In some occasions you will find your variables are extremely skewed and are far from following a normal distribution. Although OLS only assumes the error term is normally distributed, this can be a problem in a regression scenario as your variables’ values might be “compressed” and you may miss variation. This can be taken care of by applying a linear transformation to the data.

Although there are an infinite number of functions f(x) that can be used to transform a distribution, in practice only a relatively small number are regularly used. For quantitative variables one can usually rely on the “family” of powers and roots. This is also known as the “ladder of powers” which refers to a sequence of algebraic transformations that may be performed on a variable to change the shape of its distribution. The ladder function is an amazing feature, it provides statistical tests for what transformation makes the variable fit a normal distribution better. The gladder command is also useful since it provides a histogram for the most usual transformations as well as the identity version of the variable.

**Sytanx:**

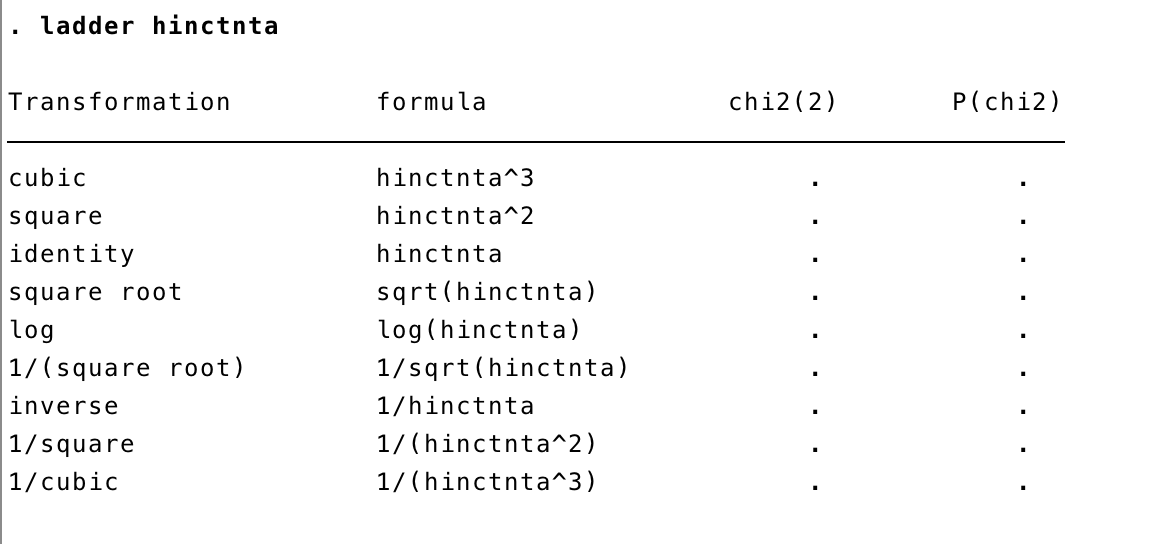
**Ladder of powers**

ladder varname [if] [in] [, generate(newvar) noadjust]

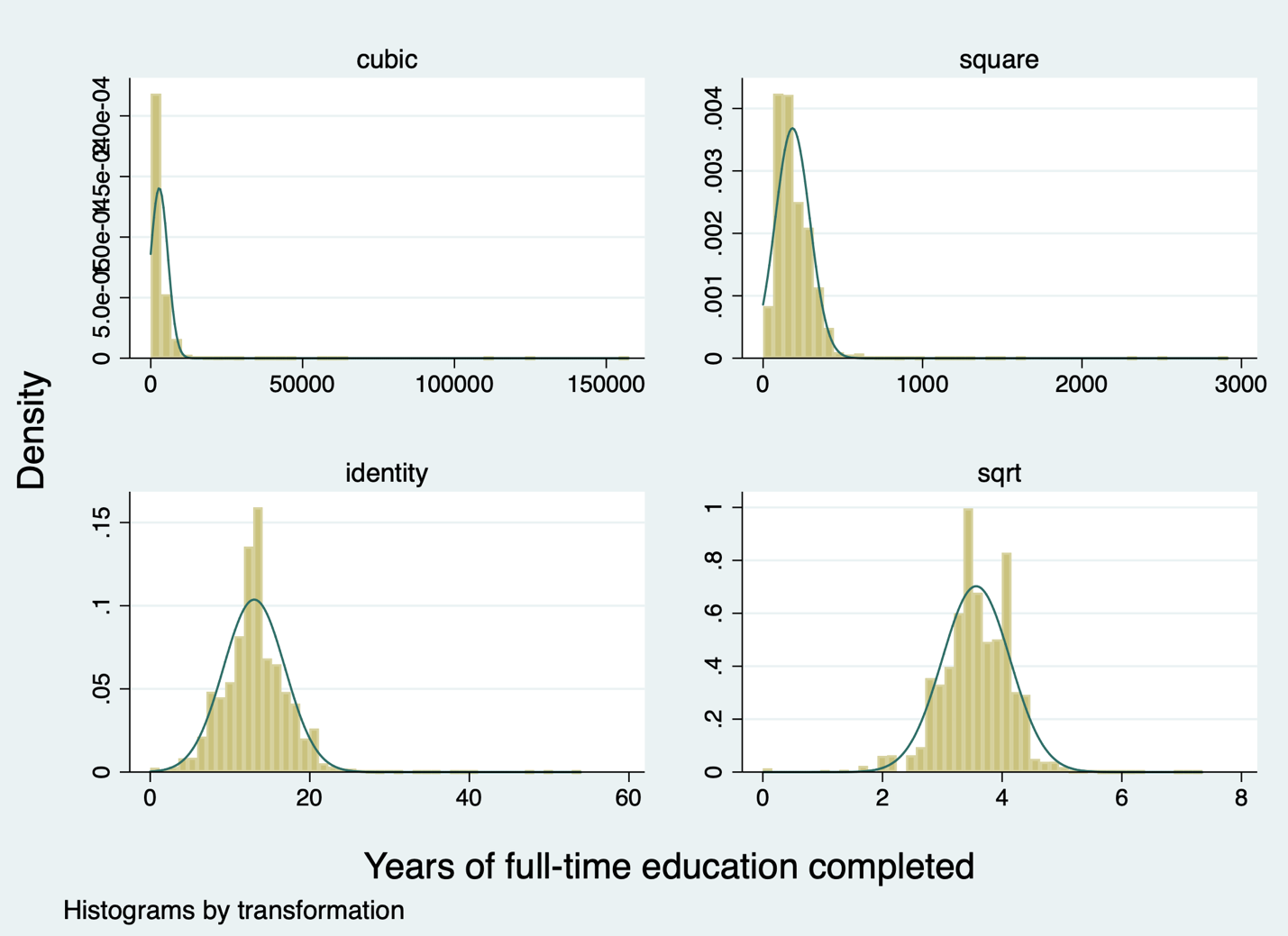
**Ladder-of-powers histograms**

gladder varname [if] [in] [, histogram\_options combine\_options]

In general you would want to choose the transformation with the smallest chi-square. In the example below, the ladder command has shown no transformation is necessary. However, if we were to find a significant statistical test for the chi2 column, we may want to consider transforming the variable.



The gladder command gives you histograms for the most common transformations and compares the distribution with the identity variable. In the example below, it doesn’t seem like any transformation really improves the distribution of the variable above the identity form.



In order to transform a variable you should use the generate command. You can generate the new variable as a function of the old one. For example:

gen log\_income = log (income)