Data Cleaning Notes

Meg Nipson

November 1, 2016

Data cleaning has two purposes:

* To acquaint you with the data and its idiosyncrasies
* To get the data ready for use in analyses

Groundwork

* Once you’ve narrowed down your research questions and identified the source of the administrative data you will need, figure out what you can do to learn more about the data. This will vary depending on whether the data is from a state longitudinal data system or data warehouse, or from district-level day-to-day operational systems.
* For SLDS data, find out as much as you can about the standard data request processes, DUAs, existing codebooks, identifiability, etc. Figure out the relationship between the agency holding the data and the state department of education, if they are different. Find out if CEPR or any of your colleagues has any contacts in or experience with the given agency.
* For administrative (e.g. district-level) data, you are likely to have to assemble data from disparate sources: student enrollment systems, class rosters, test and accountability reporting, human resources and finance, teacher evaluation data, national student clearinghouse data. Try to schedule a site visit where you can interview key data owners about the critical data elements you need, overall data coverage, quality, years available, data structure, and linking variables available. This will help you collaborate with agency staff on an extraction (“query”) design, and it will help you build relationships with contacts who can answer questions about the data during the cleaning process.

Data Acquisition

* Once you’ve laid the groundwork with the agency, gotten the necessary approvals within Harvard and the agency, and completed the IRB and DUA processes, you can acquire the data. Be patient but gently persistent throughout this process.
* For SLDS data, the data extraction formats will be fairly standard, and probably reasonably well documented.
* For operations data, the data can come in a number of formats. Beware of the temptation to let an agency analyst perform complex queries, filter the data, or clean up the data for you. If you’ll be working with the agency for a number of years, it’s likely you’ll want to re-acquire the same data later on, and your relationship with the agency will likely outlast the tenure of the person who does the extraction. Typically, you want to get the data in as close to native format as possible.

Data cleaning

* When you receive the data, it will probably be in a number of separate flat files which you will need to clean and then link together. You should start by working through each file to identify its key variables and structure (in Stata, the “duplicates report” command is helpful for this). Typically, agency data arrives in files that are nearly but not quite unique (for example, almost one record per teacher per year per school, but with a handful of anomalies).
* After you’ve identified the key variables and before cleaning the other variables, do a systematic set of test merges to make sure that the key ID variables are compatible and that you will be able to combine the files. Ideally, you’ve already discussed this issue with your agency contacts during the groundwork phase.
* Make a table with the appropriate counts for each file by year to check coverage. For example, you might want to use the unique command in State to calculate total records per year, unique students per year, and unique teachers per year. Use the IPEDS Common Core of Data or agency websites or enrollment data to check that your counts are reasonable.
* If you will be using National Student Clearinghouse data, check which colleges in your area don’t report to NSC, and estimate the potential level of undercoverage.
* If you identify problems after doing merge or count checks, reach out to your agency contacts to discuss them. Figure out together whether new data extracts will solve the problem.
* Once the merge and count checks are done, start cleaning each file individually. Ideally, you should clean all the raw data before combining files and restricting the data to your analysis sample, but you can focus your effort on the files and data elements that are most important to you.
* Starting with the raw files, write code to generate cleaned data files. Here are examples of typical data cleaning tasks:
  + Change files from “nearly unique” to “unique” at the level of their key variables by investigating the structure of the near-duplicates.
  + Tab each categorical variable, including missing values, and check its distribution in time and space (eg by school year and by a unit of geography such as school or district) to look for changes in value coding across years, incomplete coverage, or other problems.
  + Summarize numeric variables and look at the distribution of each. Check to see that values are reasonable and consistent, and investigate patterns of missingness.
  + For test score data, make histograms by test, grade, and school year. Look for changes in score ranges, illogical values, floor and ceiling effects, and changes in tests administered across time.
  + Fix inconsistent values (eg gender changing across time) by taking modal values or using other decision rules.
  + Decide how to handle outlier or illogical values. You might decide to use top and bottom coding, change them to missing, or leave them as is.
  + If you are using Stata, convert string categorical variables to numeric codes with value labels.
  + Parse complicated variables (for example, test codes) into their component parts, and clean those.
  + Incorporate external codebook information (e.g., job codes or withdrawal codes).
  + For complicated, messy files like course files, use string manipulation commands to conform course names and other variables as needed.
  + Where you have code-description variable pairs, check to see that the relationship is 1-1, or 1-1 within year (the Stata egen nvals command is useful for this), and conform the descriptions or codes as appropriate. In Stata, try to use the descriptions as labels for the code values.
  + When working with data at the school level, make sure you understand which schools are traditional schools and which are alternative. Incorporate that information into the cleaned data files. You may also want to investigate school calendars and incorporate beginning and end of school year information.
  + If you’re faced with a situation where you don’t know how to write the cleaning code that you need, first check your program’s help files, and then ask for advice. Stata and R have steep learning curves, and you can learn faster by looking at other code examples and talking to someone who has tackled a similar task.
* Ideally, you want to write code for your project that takes data all the way from raw files to analysis in a way that is replicable and readable. Here are some strategies you can use:
  + Make your program code clean and readable by using a clear structure and enough comments.
  + Document your decision rules in comments in your program file. You can also use the Stata notes command to document decision rules within the data file itself.
  + Give variables informative, detailed labels. Try to make your dataset “self-documenting” by including as much detail as possible about variable definitions and idiosyncrasies in variable and value labels.
  + Implement a consistent variable naming strategy, and put the variables in an order that makes the contents of the data file easy to see at a glance.

Building the Analysis File

* Once the raw files are cleaned, you can merge and restructure the cleaned files for analysis. You may need to reshape data from wide to long or vice versa, change transactional data to indicators by year, combine and simplify overlapping enrollment spells, or perform other manipulations. Ask for help if you’re faced with a challenge! It’s likely someone at CEPR has tackled similar challenges before.
* Once you’ve combined the cleaned files and built your analysis file, define the derived variables you need for your analyses. You might want to keep all the derived variable definitions together in one place in your code, so you can reference and add to them quickly.