Ordinal Variables and Missing Data

November 15, 2019

General

- This paper should be an intersection of ordinal variables and missing data. Jeff thinks there is room there for a contribution
- The general idea is that general treatments of missing data (listwise deletion, multiple imputation etc.) lead to different results – depending on the type of variable. How you treat `Don't Know` and `Refused` and how this treatment affects the results depends greatly on the type of variable that you use to impute the missing data. The idea is that you would need to approach things differently depending on whether you use a nominal, interval, or ordinal variable to fill in values for `Don't Know`/`Refused`. So I am developing a method to specifically treat `Don't Know`/`Refused` with ordinal variables that improves over current uses that are generic for all types of variables
- This method should be a customized form of multiple imputation that is closer to the data than general multiple imputation. Jeff and Skyler developed affinity scores and hot decking in their BJPS paper. They used the number of exact matches (in the form of other participants) to calculate the affinity score. Instead, I should use a weighted distance solution between ordinal variable categories. I would use the OP model from the blocking paper to weight the distances between the categories in matches (in the form of other participants). In other words, I would use the underlying ordered probit numbers to create the weights. So this would be a specific ordinal variable adjustment of the affinity score building
- Set up chapter structure

Code

- My code functions for one variable with NAs, for 10,000 iterations
  - I have saved results for `inc`, `age`, `Dem`, `Rep` for `hd.ord` and `hd.norm`
  - For all 4, `na.omit` performs the best. This isn't surprising, since deleting observations with missing values shouldn't be a problem with MCAR
- My code does not function When run for several variables with NAs
  - It throws up a replacement error somewhere down the line when run for 10,000 iterations. It's never in the same place, so it must be something random
  - Always the same: # of NAs overall. Not always the same between some runs: # of NAs per column, # of rows with NAs
  - Find out why my code doesn't work for NAs in several variables
    * The number of NAs inserted wasn't actually the issue. It was in the `OPMord` code
    * I painfully ran 1,000 iterations for two variables, gradually adding one line at a time to the function. I discovered the error was where I assign values to `df.cases`, specifically where the columns are all combined and the resulting vector added to the data as `educ.new`

        * I saved the vector to an empty vector, ran everything for 1,000 iterations, then looked at the vectors and the corresponding versions of `df.cases`
        * There were rows with all NAs in `df.cases`, which makes the vector shorter than the data, which means it can't be assigned to it as a column. I reran the `df.cases` creation code to find the culprit
        * I had overlooked that `int.df$Values` has one value fewer than there are variable levels, since it lists cutpoints between the levels. For 6 levels, I had assigned `int.df$length(levels(...))`, which picks the 6th value of `int.df$Values` – but it only goes to 5. I had to add `-1`. Now it's working
- Method to insert NAs
    - I tried `prodNA`, which works for `MCAR` but gives me nothing for `MAR`. And `MCAR` works well for `na.omit`, since by definition it's a random sample of missing data, which doesn't matter
    - Use `mice` `ampute()`
        * It has `MCAR` and `MAR` options. `MAR` is what I need
        * It doesn't work on just one variable; it needs at least two
        * `ampute()` works very well
- With the code adjusted (`-1`), `OPMord` now works. However, some of the `int.df`s don't have all the education levels: `int.df` with all levels has 6 rows (one fewer than the levels, since each row shows a cutoff), but some have 5. This means `OPMcut` now doesn't work
    - I could adjust `OPMcut` to work with fewer levels, but then I would be, in the end, taking means of most data with all levels and some data with fewer levels. That's problematic
    - It's better to discard the iterations where `int.df` has 5 rows after `OPMord` has run and then continue with the other functions. I set the code up to do that. However, after about 40 percent of running 12,500 iterations, there was an error: I hadn't factored in that the `int.df`s could also have fewer rows than that. A few of them had 4 rows, which caused the error. I adjusted the code to now discard the iterations where `int.df` has anything other than 6 rows
- The data is currently running for 12,500 iterations, 80 percent NAs, `Rep` and `inc`, `hot.deck.ord`, `hot.deck.norm`, `na.omit`
- Add more variables/NAs
    - Add every demographic to the data that I still have: Age(numeric), employment (5 levels), ideology (liberal, conservative, neither), following public affairs (ordinal, 4 levels), media interest (numeric, accumulative count of activities), participation (numeric, accumulative count of activities)
    - Add NAs to 5 of the variables
- More methods
    - Include mice imputation
- What to run `hot.deck.norm` on
    - I currently run it on the data with the cutpoints as the education values
    - Run it on the original education values as well
- Percentages of NAs
    - I'm currently using 80 percent. Jeff used 20, 50, and 80

  - Expand to 20 and 50 percent NAs when suitable
- `sdCutoff` in `hot.deck.norm`
  - The default of `sdCutoff` is set to 10. Education has 7 levels, so it won't be considered continuous and thus won't be scaled down. This holds for either data– the original one or the one with the cutpoints. A lot of the variable data will be then be ignored since it doesn't fall in the region of 0 or 1
  - This feels like a tuning parameter to Jeff. Try shorter test runs with different cutoffs, to possibly rule out ridiculous choices

Theory

- Step by step fill in the sections on Missing Data, Deletion, and Imputation
- Rework the introduction
  - Current stuff in there is very broad and nowhere near detailed enough (taken from the Kerwin application)