final ps/final project; due may1

[version: Wednesday 16th April, 2025 10:50]

1. build on ps4 (see ps4 instructions again): fix it up and beef it up
2. may use some of the bonus libs like plotly, html/website hosting; but better have solid story just with mpl than fancy meaningless stuff; again: idea and story matter most and simplicity first
3. now it's the time to take a step back and think hard about your vis: what it all means, what's the story there, give a lot of thought to interpretations, be critical; and also circle back to initial research questions and hypothesis–found evidence in support or against? strong evidence or weak? and maybe conditional? say yes for female, but no for males, etc
4. list problems/limitations: do have a section on limitatations/future res

---

general directions (always the same):

- i will show your code in class and possibly post some of your code or link to it–again, as per our core values–opensource, transparency, sharing; but if you'd like to keep your code private, that's fine–just let me know, and i will keep your code secret (no penalty, except that you may get less feedback–if we discuss your code in the class, you will benefit from it!)
- you must submit all the code that was executed from the very beginning starting with the very raw data as per replication principle; unless data is too big to fit online, then just start with a comment, eg "to fit data online i had to take a random sample of 10perc"
- all ps are mostly cumulative–you can, and should, include much of previous code you've written for this class; can also use code you've written outside of this class (other classes, projects, etc)–but you have to clearly mark the code that has not been written for this class–otherwise, scholastic dishonesty!
- because you are only submitting code, it must load data from Internet–most data already online; if not: put your data into your github, google drive, etc; (when you put data into any public space, try not to violate data copyrights... I haven't heard of anyone having problems with that, but be careful–for instance you may subset dataset to few vars and smaller sample); and it is also easier to experiment on small datasets
- keep it simple! at the beginning of your notebook drop unnecessary vars; and even retain only certain, say most important, observations; keep it manageable; it is much easier to learn using simple data; can always complicate later!; much better to do it right using simple data than do it wrong using complex data!
- have nice structure in your file: sections, subsections, etc; may also have multiple files
- can submit ps early and email listserv and ask for comments
- it is great to copy code from others; again, one of the rules for this class is 'be lazy': don't reinvent the wheel, whatever you are coding, it has already been done, google things often; but of course you cannot submit 100% code by someone's else; and if you submit a substantial chunk not written by you and for this class, cite!
- if you do something extra/fancy that is relevant and closely related to the assignment questions, it will be extra credit
- use coding rules that we've learned so far
- submit (only) the Py notebook into git repo; ps are due by the beginning of the next class unless indicated otherwise, eg "due in 2 weeks"; late ps are not accepted
- we are on the way to developing the final project with these ps: as we progress, your ps should start resembling a coherent and logical project where you use learned techniques to answer interesting questions– say in few sentences (probably at the beginning) why are you doing what you are doing–that is, answer the "so what question": "ok, you're gonna run all that code, and so what?" what's the goal of all that, why are you doing this? you need a compelling justification for what you are doing; typically: to answer some exciting questions: say what are those questions you want to answer; related: say why you use data you are using, is it best?, does it serve the purpose?; also, feel free to ask us questions in comments