Human Computer Interaction

Victor Møller Poulsen, Studie Nr.: 201707639

June 1, 2021

1 Introduction

2 Target group

Most specifically students at cognitive science who have learned bayesian statistics in R/brms/stan but would like to transition to python/pymc3/theano, or simply to learn how to conduct a bayesian analysis in a second coding language. The workflow attempts to blend what has been taught by Riccardo in the course computational modeling for cognitive science, the workflow proposed by Gelman (ref) and what is recommended as best practise in various pyMC3 tutorials (and the like). While cognitive science students are my main demographic, this is a very narrow target group. The notebook should be general enough that everyone who is looking to transition from R/brms to python/pyMC3 should find the webpage

useful.

Python is developing rapidly in data science and statistical analysis, and while R has been dominant for specifically statistical analysis this is not so clear anymore (ref??).

As I have tried to highlight in the webpage (product) I find that certain things are easier in R/brms and certain things are easier in python/pyMC3. One thing that I find easier in python/pyMC3 is doing customized analysis. This is because R/stan relies on the easy-to-use model formula that is well known from lme4 (see image). In Python/pyMC3 this has also been developed as part of the glm module (right?) but the usual way is to explicitly connect the building blocks (distributions) manually within a model context. Besides allowing for easy customizability this also ensures that we actually understand how our model works and what we are doing, which I think is positive.

3 What we have to include (notes)

1. Get predictions to work

2. Draw our own Gelman

4 Good points we need to remember

usability vs functionality, e.g.

1. pop-out (e.g. the think deeper sections)

2. not presenting all priors (not that important), but for those interested they can get it. Does not clutter the page.

3. Focus on the target group, e.g. trying to make the python format as easy as possible while still remaining true to the format. E.g. avoid object oriented programming in my code-base, work as much as possible with data frames rather than raw vectors (which is typical in pyMC3 actually). Trying to make the material easy to digest for the target group while not compromizing with functionality (of the code). They can dive deeper on their own after this introduction.

4. streamlit is most used in datascience and machine learning (ML) with models that fit really fast (i.e. ¡ 1 second). streamlit presentations that rely on models of this kind can let the user freely play around with models, variables, etc. and fit the models in real time, thus making them really interactive. Since we cannot sample Bayesian models within a reasonable time-frame (i.e. it takes ¿ 30 seconds) I had to think of alternatives to make the streamlit app feel interactive without requiring the user to wait excessively long. This was done by pre-rendering (bla, bla, bla ... leads to code base in next section).

Design, e.g.

1. Difference between pop-out (e.g. think deeper) and the buttons (e.g. prior levels).

2. Nice color theme, to get them interested.

3. Visually oriented. Focus on figures for the main page, and then a lot of the code and writing in optional boxes. Focus on just showing the workflow (keeping it simple) and then referring them to other more ”deep” places.

5 Code base & Reproducibility

A lot of the development time for this project was spent ensuring a solid code-base. This was necessary for two reasons.

First, because I had to pre-compile a lot of models and plots, I had to make sure that my code was reproducible, without bugs, and was possible to navigate. This was necessary from a developmental point of view (i.e. it was necessary for me).

Second, it also means that the naming schemes for everything - across R and Python and all parts of the analysis is consistent, and that it should be easy for users to navigate the code-base. I.e. those users who want to look deeper than just the streamlit app and actually check the github.

6 Conclusion