Peer review for Jingjing Fan

Tyler Buffington

March 15, 2019

Hi Jingjing!

First of all, I really appreciate how your written notes are so thorough, explicit, and well organized. From what I saw, it is really easy to follow the steps you took to solve the problems. On top of that, the use of different colors also helps helps create a visual distinction between the problem statement, your derivations, and your "aside" comments. Regarding your code, I'll start by prefacing that I am much more experienced in python than R, but I found it pretty easy to follow what you did despite that.
Before I get into coding feedback, I want to offer some brief comments regarding GitHub usage. I noticed that many of your commit messages are "add files via upload." I would highly recommend using informative commit messages even if they're really concise. Something like "added exercises 2" or "revised cross validation code" can be really helpful for tracking the history of changes if you ever need to do that. It's probably not necessary for the purposes of this class, but getting into that habit is useful if you ever have to use GitHub for collaborative coding projects. Also, it looks like you have been updating the files on GitHub by manually deleting old versions and then uploading newer versions manually though the website. If this is the case, I would recommend setting up a local clone of your repository. The general idea is that you would have a folder on your computer that matches your online repository. Then when you make changes to your code (or other files) on your computer, you just need to commit and then push them and they will show up on the website. I personally like to use the command prompt to do this, but you can also use GitHub desktop. If you have any questions, let me know!
Given that the main purpose of the peer reviews is to receive coding feedback, I will structure the rest of this document by offering my comments on each of the main coding tasks that we have done through local polynomial regression.

1 Bayesian linear model comments

My main recommendation for this section would be to use a function that returns the fit parameters for a given dataset. This would make your code more modular so that you could easily use it later for a different problem. Even for the purposes of the exercise, many of the lines of code you have written show up twice because you have two cases- the "uninformed" prior and the "informed" prior. If these lines of code lived inside of a function, you would reduce the total lines of code because you could just call that function twice with two different sets of arguments. Besides that, I think that the organization looks good and the code is pretty well commented. The only place I would recommend additional commentation would be on the line where you add a column of ones to your x data. Including a comment like "Appending a column of ones so the model will fit an intercept" would be helpful to someone who isn't familiar with your code.

2 Heavy tailed error model

My previous comment about packaging your code in a function is also relevant here. I think it also would be helpful to describe what each of your hyperparameters are and which data type they are (e.g. float, integer, array, etc.). I think it would be good to also use comments to explain the difference between n and N, but I eventually realized that n is the number of samples you keep, whereas N is the number of samples that you iterate over. Also, it would be helpful to make your values for burning and thinning variables rather than hardcoding them. Also, I'm sure you completed this code prior to our lcass discussion, but I think it's worth reiterating James' comments about how thinning isn't advantageous unless memory is a concern.

3 Kernel smoothing

The functions look great! I really like how you specify the data type of the inputs and outputs of your function. This allowed me to understand very quickly what you did. I think the commenting is also clear and well organized. My only recommendation would be to make it clear in the function description that KernWeights returns normalized weights.

4 Cross validation

My main recommendation here would be to make it so that your crossVal function returns the a cross validation score, rather than an optimal bandwidth parameter. This would allow the function to be used in applications beyond the scope of the course exercises. Also, the crossVal function seems to split the training set into two groups of equal size, but it would be good to allow the user to specify the size of the training/test sets through an input argument. As far as determining the optimal h, I would recommend the use of an optimization function rather than evaluating the best bandwidth out of an array of potential candidates. I believe the optim() function in R would be useful here, but take that with a grain of salt from a python user.

5 Local polynomial regression

Overall, this looks good! My only comment would be to make a function that returns the fitted values, rather than the hat matrix. This is because the hat matrix is a quantity that is used for an intermediate step, and its interpretation is far less clear to potential users than the fitted y values. You could make the hat matrix an optional additional return argument if you want to use it for other analyses.