SDS385 Peer evaluation 1:

Reviewee: https://github.com/JKelle/SDS-385

This is a summary of the suggestions I discussed with you during the meeting we had on Sept. 14.

1. **Code efficiency**.

    a. My suggestion is to use R's built-in function *crossprod* whenever you can. It can help preventing from doing a lot of useless computation, especially in the case of symmetric matrices. It is evident in the weighted least squares problem.
    b. Apart from that, we used the same tricks to multiply a diagonal matrix by a full matrix, which explains why our benchmarked performances look pretty similar and fast.

2. **Functions structure**.

    a. In my experience, it is a good programming rule to feed the functions all of the parameters of a given problem. In particular, I suggest you to feed it also with the tolerance threshold and with the maximum number of iterations instead of defining those variables inside the *gradient* function. In this way you can "play" with those parameters to see if anything changes.

3. **Check performance**.

    a. Apart from checking the convergence of the log-likelihood, I would also compare the final regression parameters with the ones obtained by R's built-in *glm* function. In this way you have a benchmark you can use to compare the results given by several algorithms.
    b. As far as the convergence check is concerned, you can use threshold on the relative decrement of the log-likelihood function, that is, *(lik.prec – lik.act) / lik.prec*.
    c. I like the idea of using the accuracy of the logit classifier as a performance comparison between those algorithms. They are likely to give the same result, but this is a good way to check if the logit model is appropriate.

4. **Other comments.**

    a. You compute the vectorized version of the log-likelihood, which is the most efficient version.
    b. You already implemented line search methods for the gradient descent, which is great!