Maximum Entropy

Yang Liu

Last night I found that the implementation of Maximum Entropy I did is not corresponding to the formulas given. So I wrote a new one, hope this would be a right one.

I was misled by the data from homework3 and the ppt (since the x vector there do not have a arrow above or have a bold font). I used to hold the idea that each word and label pair is a data, which is implemented in the version one.

The biggest difference between this one and previous one is when computing log likelihood and gradient. When a blog is a data, which means x is vector that holds every word in the blog, the numpy array could be utilized to simplify the computation.

I store every P(y|**x**) in a dictionary so that if the same data pair wants its P, it could be provided instead of computed again.

When eliminating some of the words that appears too many and too few times, the performance would not decrease a lot but the time used could be shorted obviously. I implemented this method in eliminate_features() by deleting the words in both the blog data that needed to compute log likelihood and gradient and the feature data that stores each kind of feature.

With names examples, the accuracy would 79%. With blogs, when all the words take into account, the scores would be: accuracy: 0.67 precision: 0.71 recall: 0.69 F1: 0.70, and time is 215s. When words that appears less than 200 times and more than 800 times are eliminated, the scores would be: accuracy: 0.68 precision: 0.71 recall: 0.71 F1: 0.71, time is 64s.

Last, even though the results are good (maybe just coincidence?), there is problem existed in my program but I don't have enough to debug. When doing the name example, everything is good. But when it comes to the blog, the optimizer would not end properly, there would be have an overflow. I believe much time of total time is wasted in computing useless log likelihood and gradients. I'm so appreciated if you could help me figure out where the problem is.