Assignment #3

Due: 23:59pm October 20, 2013

Complete two of the following three questions.

Each of these questions is somewhat open-ended and uses the same data set. These questions are meant to be representative of the kind of decisions you might have to make in modeling real data. That is to say that you have have to make decisions along the way on how to fill in the gaps about how to handle missing data, assessing convergence, etc. All three problems use a slightly perturbed and preprocessed variant of the Jester collaborative filtering data (version 2) from `http://eigentaste.berkeley.edu/dataset/`, which you also used in the first assignment. These data are user ratings of 150 jokes. There are over 1.7M ratings in the range $(-10, 10)$, from about sixty thousand users. As the data are slightly perturbed, please use the version at `http://www.seas.harvard.edu/courses/cs281#jester`. The texts of the jokes are also available at this URL. The correct reference for these data is: Ken Goldberg, Theresa Roeder, Dhruv Gupta and Chris Perkins. *Eigentaste: A Constant Time Collaborative Filtering Algorithm*. Information Retrieval 4(2):133-151, 2001. Warning: most of the jokes are bad, tasteless or both. At best, they were funny to late-night TV hosts in 1999-2000. Note also that some of the jokes do not have any ratings and so can be ignored.

It is not strictly necessary to use the Jester data. If there is another equivalent data set that you are interested in and for which it is possible to answer questions that are comparable in spirit and difficulty, you are welcome to use that data set instead. If you are not sure whether the data you have are appropriate, feel free to contact the course staff.

Problem 1 (50pts)

Clustering Jokes and Ratings with Expectation Maximization: The objective of this problem is to model the joke ratings via a mixture model.

1. Use expectation maximization to fit a mixture of rescaled beta distributions to the overall ratings distribution. (Recall that in assignment one, you computed the MLE for a rescaled beta distribution.) Unlike the case with mixture of Gaussians, you'll probably need to perform some explicit optimization within the maximization step.

   (a) Run several different random restarts. Do you get the same final log likelihoods? About how many restarts do you need to do to get a consistent answer? (You might try different initialization techniques, too.)

   (b) Choose a metric for determining the number of mixture components, e.g., cross validation. Now evaluate different numbers of components and produce a bar chart or box plot that helps you decide how many to use.

   (c) Having chosen an appropriate number of components, produce a plot that shows the expected complete data log likelihood as a function of EM iteration.

2. It's not very satisfying to just build a model for the marginal distributions over ratings, because it doesn't tell us anything about the differences between jokes. Let's now build a model that clusters the jokes and use EM to fit it.

   (a) The real-valued ratings are a bit unwieldy to deal with in the first pass. Quantize the ratings into an ordinal set, with perhaps ten categories. You can do this by simple binning, by using your rescaled-beta model from above, or some other method. Explain what you did.

   (b) Having quantized the ratings, normalize them for each joke so that they are vectors on the simplex. Make sure you handle any cases where you have zero counts in your quantization. Each joke now has a "feature vector" that we can model as coming from a Dirichlet distribution. Derive a procedure for finding the maximum likelihood estimate of a Dirichlet distribution, given a set of such vectors. Describe your procedure.

   (c) Implement a mixture of Dirichlet distributions for clustering the jokes. Use expectation maximization and, as in the case above, explain your procedure for selecting the number of components and produce a graphical representation of the quality of different choices.

   (d) Examine the clusterings. Did the model discover anything interesting in the way that people rate the jokes?

3. In an ideal world, we would go farther and combine these models so that each joke provided a joke-specific set of mixture weights for the rescaled betas. That is, each joke would produce a unique weighting drawn from its cluster-specific Dirichlet distribution, and rather than crudely quantizing things we would use that in the rescaled beta mixture. Draw a directed graphical model that reflects such a structure and explain your notation and priors. Note that plate notation can be nested and overlapped to make things clearer.

Problem 2 (50pts)

GLM Regression of Ratings with Text Features: The approach above did not use the text of the jokes at all. However, the joke text should of course be the most important thing in determining a rating. Let's build a generalized linear model using the rescaled beta distribution. This will be a variant of *beta regression* and there are different ways we might imagine tackling it (Google Scholar gives a lot of results for "beta regression").

1. Decide on a feature representation for the jokes. The easiest thing is probably to compute a binary representation that indicates the presence or absence of, say, the fifty most common words. We've provided a file that has the joke texts with punctuation and case removed in `jester_items.clean.dat`. This means you'll need to do a bit of text processing to turn these into features that you can use in the regression. Also, you are welcome to try different and more interesting things: different numbers of words, counts instead of binary features, tf-idf, remove stop words, etc. Explain what you did and why.

2. Formulate an approach to rescaled beta regression. Describe your parameterization and your link function.

3. Compute the gradient of the log likelihood in terms of the weights. Be sure to verify the gradient using finite differences.

4. Place a Gaussian prior on the weights and use stochastic gradient descent with mini-batches of data to fit the MAP regression weights. Evaluate different prior variances using, e.g., cross validation. Produce a bar chart or box plot that shows the different values you tried. Also describe what mini-batch size and learning rates seemed to work best.

5. Having computed MAP weights, go back into your feature representation and examine which words seem to have the biggest influence on the resulting rating, as determined by magnitude of the weights. Find anything interesting?

6. Try some kind of interesting tweak. For example, use a different feature representation or use ℓ_1 regularization instead of ℓ_2. Do your results change in any interesting ways? Describe what you decided to do.

Problem 3 (50pts)

Modeling Users and Jokes with a Latent Linear Model: Neither of the previous problems tackled any modeling of users. Of course, different users will have different tastes and different rating calibrations. In this problem, we'll use a latent linear model to jointly model users and jokes.

1. Let $r_{i,j} \in (-10, 10)$ be the rating of user i on joke j. A latent linear model introduces a vector $\boldsymbol{u}_i \in \mathbb{R}^K$ for each user and a vector $\boldsymbol{v}_j \in \mathbb{R}^K$ for each joke. Then, each rating is modeled as a noisy version of the appropriate inner product. For example:

$$r_{i,j} \sim \mathcal{N}(\boldsymbol{u}_i^\mathsf{T} \boldsymbol{v}_j, \sigma^2). \tag{1}$$

   Start out with a small K, e.g., $K = 3$, and set this up as a maximum likelihood estimation problem. Derive gradients for the user and joke vectors, as well as the noise variance, and implement learning. It will probably be necessary to use stochastic gradient descent with a bit of cleverness in updating user parameters. That is, only a small number of users will appear in each mini-batch. What variance did you find? Also, investigate the extremes of the different latent dimensions. Can you identify interesting properties of jokes that are captured by the "principal directions" of the latent space?

2. Come up with a quantitative metric for evaluating dimensionality of the latent space, e.g., cross validation. Evaluate different K and produce a bar chart or a box plot that shows the associated performance. What seems like a good value for K?

3. We might imagine that some jokes are just better or worse than others. We might also imagine that some users tend to have higher or lower means in their ratings. Introduce such biases into the model and fit it again, learning these new biases as well. Explain how you did this. One side-effect is that you should be able to rank the jokes from best to worst. What are the best and worst jokes?

4. Sometimes we have users or jokes that only have a few ratings. We don't want to overfit with these and so we might want to put priors on them. What are reasonable priors for the latent features and the biases? Draw a directed graphical model that shows all of these variables and their relationships.

5. Extend your model in some way and see if you discover any more interesting structure. Some ideas: make your model heteroscedastic and have the variance change as a function of the latent factors; use the beta GLM approach of the previous problem for the likelihood, rather than a Gaussian; implement the priors you chose above and see if you get better or worse results; come up with a way to incorporate the text features you used in the previous problem. Explain what you did and why.