CSE 515: Statistical Methods in Computer Science

Homework #1

Due at noon on January 28th

Guidelines: You can brainstorm with others, but please solve the problems and write up the answers by yourself. You may use textbooks (Koller & Friedman, Russel & Norvig, etc.), lecture notes, and standard programming references (e.g., online Java API documentation). Please do NOT use any other resources or references (e.g., example code, online problem solutions, etc.) without asking.

Submission instructions: Submit this assignment by email to Chloé Kiddon (chloe@cs). Attachments should include: A PDF containing written answers; source code for the mixture model; and a README explaining how to compile and run the source code under Linux (e.g., tricycle).

1. **Disease testing** A rare disease – PhDitis – affects 1% of the population. A certain test for this disease – based on how fast a person can find free food in a closed environment – is 95% effective at determining if a person has the disease. You decide to get tested and your test, unsurprisingly, comes back positive. What is the probability that you actually have the disease?

2. **Probability Theory** For random variables $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{Z}$, if $\mathbf{X}$ is conditionally independent of $\mathbf{Y}$ given $\mathbf{Z}$, we denote this by $(\mathbf{X} \perp \mathbf{Y} \mid \mathbf{Z})$. See section 2.1.4 of Koller & Friedman for more on conditional independence. Note that $P(\mathbf{X} \mid \mathbf{Z} = z)$ is undefined if $P(\mathbf{Z} = z) = 0$.

   (a) Prove that each of the following two properties hold for any probability distribution P.

   Weak Union:
$$(\mathbf{X} \perp \mathbf{Y}, \mathbf{W} \mid \mathbf{Z}) \Longrightarrow (\mathbf{X} \perp \mathbf{Y} \mid \mathbf{Z}, \mathbf{W})$$

   Contraction:
$$(\mathbf{X} \perp \mathbf{W} \mid \mathbf{Z}, \mathbf{Y}) \ \ \& \ \ (\mathbf{X} \perp \mathbf{Y} \mid \mathbf{Z}) \Longrightarrow (\mathbf{X} \perp \mathbf{Y}, \mathbf{W} \mid \mathbf{Z}).$$

   (b) Prove that the Intersection property holds for any positive probability distribution P. You should assume that $\mathbf{X}$, $\mathbf{Y}$, $\mathbf{Z}$, and $\mathbf{W}$ are disjoint. The Intersection property states that
$$(\mathbf{X} \perp \mathbf{Y} \mid \mathbf{Z}, \mathbf{W}) \ \ \& \ \ (\mathbf{X} \perp \mathbf{W} \mid \mathbf{Z}, \mathbf{Y}) \Longrightarrow (\mathbf{X} \perp \mathbf{Y}, \mathbf{W} \mid \mathbf{Z}).$$

(c) Provide a counter-example to the Intersection property in cases where the distribution P is not positive.

3. **Dirichlet priors** In this problem you will show that the family of mixtures of Dirichlet priors is conjugate to the multinomial distribution.

   (a) Consider a simple possibly-biased-coin setting. Assume we use a prior which is a mixture of two Dirichlet (Beta) distributions: $P(\theta) = 0.95 \cdot Beta(5000, 5000) + 0.05 \cdot Beta(1, 1)$; the first component represents a fair coin (for which we have seen many imaginary samples), and the second represents a possibly-biased coin, whose parameter we know nothing about. (1) Show that the expected probability of heads given this prior (i.e., the probability of heads averaged over the prior) is 1/2. Now suppose we observe the data sequence $(H, H, T, H, H, H, H, H, H, H)$. (2) Calculate the posterior over θ, $P(\theta \mid D)$. (3) Show that it is also a 2-component mixture of Beta distributions by writing the posterior in the form $\lambda^1 Beta(\alpha_1^1, \alpha_2^1) + \lambda^2 Beta(\alpha_1^2, \alpha_2^2)$. Provide actual numeric values for the different parameters $\lambda^1, \lambda^2, \alpha_1^1, \alpha_2^1, \alpha_1^2, \alpha_2^2$.

   (b) Now generalize your calculations from part (a) to the case of a mixture of d Dirichlet priors over a k-valued multinomial parameter. More precisely, assume that the prior has the form:

   $$P(\theta) = \sum_{i=1}^{d} \lambda^i Dirichlet(\alpha_1^i, \dots, \alpha_k^i)$$

   and prove that the posterior has the same form.

4. **Programming project** Implement the EM algorithm for mixtures of Gaussians. You can use C, C++, Java, or Python. Assume that means, covariances, and cluster priors are all unknown. For simplicity, you can assume that covariance matrices are diagonal (i.e., all you need to estimate is the variance of each variable). Initialize the cluster priors to a uniform distribution and the standard deviations to a fixed fraction of the range of each variable. Your algorithm should run until the relative change in the log likelihood of the training data falls below some threshold (e.g., stop when log likelihood improves by < 0.1%). The program should be run on the command line with the following arguments:

```
./gaussmix <# of cluster> <data file> <model file>
```

   It should read in data files in the following format:

```
<# of examples> <# of features>
<ex.1, feature 1> <ex.1, feature 2> ... < ex.1, feature n>
<ex.2, feature 1> <ex.2, feature 2> ... < ex.2, feature n>
...
```

And output a model file in the following format:
<# of clusters> <# of features>
<clust1.prior> <clust1.mean1> <clust1.mean2> ... <clust1.var1> ...
<clust2.prior> <clust2.mean1> <clust2.mean2> ... <clust2.var1> ...
...

Train and evaluate your model on the Wine dataset, available from the course Web page. Each data point represents a wine, with features representing chemical characteristics including alcohol content, color intensity, hue, etc. We provide a single default train/test split with the class removed to test generalization. You can find the full dataset and more information in the UCI repository (linked from the course Web page). Start by using 3 clusters, since the Wine dataset has three different classes. Evaluate your model on the test data.

Two recommendations:

- To avoid underflows, work with logs of probabilities, not probabilities.
- To compute the log of a sum of exponentials, use the "log-sum-exp" trick:

$$\log \sum_i \exp(x_i) = x_{max} + \log \sum_i \exp(x_i - x_{max})$$

Answer the following questions with both numerical results and discussion.

(a) Plot train and test set likelihood vs. iteration. How many iterations does EM take to converge?

(b) Experiment with two different methods for initializing the mean of each Gaussian in each cluster: random values (e.g., uniformly distributed from some reasonable range) and random examples (i.e., for each cluster, pick a random training example and use its feature values as the means for that cluster). Does one method work better than the other or do the two work approximately the same? Why do you think this is? (Use whichever version works best for the remaining questions.)

(c) Run the algorithm 10 times with different random seeds. How much does the log likelihood change from run to run?

(d) Infer the most likely cluster for each point in the training data. How does the true clustering (see **wine-true.data**) compare to yours?

(e) Graph the training and test set log likelihoods, varying the number of clusters from 1 to 10. Discuss how the training set log likelihood varies and why? Discuss how the test set log likelihood varies, how it compares to the training set log likelihood, and why. Finally, comment on how train and test set performance with the "true" number of clusters (3) comapres to more and fewer clusters and why.