Homework Assignment 1

Due: January 26, 2017 at 11am

Total points:	100
Deliverables:	`hw1.pdf` containing typeset solutions to Problems 1-3.
	Source code containing your implementation for Problem 4.
	`README` file explaining how to compile and run the source code on Linux or Windows.

Guidelines: All files must be submitted by Dropbox. You can brainstorm with others, but please solve the problems and write up the answers and code by yourself. You may use textbooks (Koller & Friedman, Russell & 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.

1 Probability

1. *(8 points)* After your yearly checkup, the doctor has bad news and good news. The bad news is that you tested positive for a serious disease, and that the test is 98% accurate (i.e., the probability of testing positive given that you have the disease is 0.98, as is the probability of testing negative given that you don't have the disease). The good news is that this is a rare disease, striking only 1 in 10,000 people. What are the chances that you actually have the disease? (Show your calculations as well as giving the final result.)

2. *(18 points)* Suppose you are given a bag containing n unbiased coins. You are told that $n-1$ of these coins are normal, with heads on one side and tails on the other, whereas one coin is a fake, with heads on both sides.

   (a) Suppose you reach into the bag, pick out a coin at random, flip it and get a head. What is the (conditional) probability that the coin you chose is the fake coin?

   (b) Suppose you continue flipping the coin for a total of k times after picking it and see k heads. Now what is the conditional probability that you picked the fake coin?

   (c) Suppose you wanted to decide whether the chosen coin was fake by flipping it k times. The decision procedure retures *fake* if all k flips come up heads; otherwise it returns *normal*. What is the (unconditional) probability that this procedure makes an error?

2 Conditional Independence

3. *(24 points)* Let P be a probability distribution and let $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{W}$ be random variables. $\boldsymbol{X}$ and $\boldsymbol{Y}$ are *conditionally independent given* $\boldsymbol{Z}$ *in* P if and only if $P(\boldsymbol{X}, \boldsymbol{Y}|\boldsymbol{Z}) = P(\boldsymbol{X}|\boldsymbol{Z})P(\boldsymbol{Y}|\boldsymbol{Z})$. This is denoted as $(\boldsymbol{X} \perp \boldsymbol{Y}|\boldsymbol{Z})$. For more on conditional independence see section 2.1.4.3 of Koller and Friedman. Prove the following properties or give a counter-example.

   (a) **Weak Union:**

$$(\boldsymbol{X} \perp \boldsymbol{Y}, \boldsymbol{W}|\boldsymbol{Z}) \Longrightarrow (\boldsymbol{X} \perp \boldsymbol{Y}|\boldsymbol{Z}, \boldsymbol{W})$$

   (b) **Contraction:**

$$(\boldsymbol{X} \perp \boldsymbol{W}|\boldsymbol{Z}, \boldsymbol{Y}) \;\&\; (\boldsymbol{X} \perp \boldsymbol{Y}|\boldsymbol{Z}) \Longrightarrow (\boldsymbol{X} \perp \boldsymbol{Y}, \boldsymbol{W}|\boldsymbol{Z})$$

(c) **Intersection:** For positive distribution P and mutually disjoint sets $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{W}$ (A distribution P is said to be *positive* if for all events $\alpha \neq \emptyset$, we have that $P(\alpha) > 0$):

$$(\boldsymbol{X} \perp \boldsymbol{Y} | \boldsymbol{Z}, \boldsymbol{W}) \ \& \ (\boldsymbol{X} \perp \boldsymbol{W} | \boldsymbol{Z}, \boldsymbol{Y}) \Longrightarrow (\boldsymbol{X} \perp \boldsymbol{Y}, \boldsymbol{W} | \boldsymbol{Z})$$

(d) Provide a counterexample to the intersection property in case where the distribution P is not positive.

3 Programming: EM for Mixtures of Gaussians

4. *(50 points)* Implement the EM algorithm for mixtures of Gaussians. You can use C, C++, Java, Python or MATLAB. 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 clusters> <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.train`) 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) compares to more and fewer clusters and why.