CS 181 Assignment 3: Clustering and Parameter Estimation

Professor David C. Parkes

Out Monday February 28th
Due Noon Friday March 11th

General Instructions:

You may work with one other person on this assignment (or you may work alone if you prefer). Each group should turn in one writeup. This assignment consists of a theoretical component and an experimental component.

Problem 1

[**15 Points**] This question investigates the use of instance-based methods such as the HAC clustering algorithm in high dimensional spaces.

(a) [**4 Points**] Let X be the m-dimensional unit hypercube, i.e., $X = \{(x_1, \ldots, x_m) : 0 \leq x_j \leq 1\}$. Let $\mathbf{x}$ be the point at the center of X, and $\mathbf{y}$ a point in X drawn randomly from a uniform distribution. For $\epsilon \in (0, \frac{1}{2})$, compute the probability ρ that $\max_j |x_j - y_j| \leq \epsilon$.

(b) [**2 Points**] Let $\mathbf{x}$ be some other point in X, and $\mathbf{y}$ a point in X drawn randomly from a uniform distribution. Based on your result from (a), argue (informally is OK) that the probability that $\max_j |x_j - y_j| \leq \epsilon$ is *at most* ρ.

(c) [**2 Points**] Let $d(\mathbf{x}, \mathbf{y})$ denote the Euclidean distance between two points $\mathbf{x}$ and $\mathbf{y}$. Prove that $d(\mathbf{x}, \mathbf{y}) \geq \max_j |x_j - y_j|$. Given this, argue from (b) that if $\mathbf{x}$ is any point in X, and $\mathbf{y}$ is a point in X drawn randomly from a uniform distribution, then the probability that $d(\mathbf{x}, \mathbf{y}) \leq \epsilon$ is also at most ρ.

(d) [**4 Points**] Using your result from (c), give a *lower bound* on the number n of points needed to guarantee, with probability at least $1 - \delta$, that the nearest neighbor of a point x will be within a radius ϵ of it.

(e) [**3 Points**] What can you conclude about the effectiveness of the HAC clustering algorithm in high dimensional spaces?

Problem 2

[**15 Points**] This question asks you to explore the maximum likelihood (ML), maximum *a posteriori* (MAP), and full Bayesian (FB) approaches to estimating the parameters of probability models.

(a) [**5 Points**] Adopting standard notation from class $\mathbf{D}, \theta, P(\Theta = \theta), P(\mathbf{D} \mid \Theta = \theta)$ and $P(\Theta = \theta \mid \mathbf{D})$, provide the probability distribution $P(\mathbf{X} = \mathbf{x} \mid \mathbf{D})$ that is learned by each of the ML, MAP and FB methods, where $\mathbf{X}$ is a random variable defining a feature vector.

(b) [**2 Points**] Explain why the MAP method can be considered to be Bayesian while the ML method cannot.

(c) [**2 Points**] Consider the MAP method. Provide a practical advantage it enjoys, as a machine learning approach, over the ML method and another that it enjoys over the FB method.

(d) [**4 Points**] Consider the soccer team example first introduced in the class notes for lecture 10 and then revisited in the notes for lecture 11. Sketch or plot the Beta distribution for Beta(1,1), Beta(3,3) and Beta(2,5). Explain the intuition that is represented by adopting each of Beta(1,1), Beta(3,3) and Beta(2,5) as priors for the probability of a win.

(e) [**2 Points**] Why is the Beta distribution useful when used together with MAP estimation for reasoning with Boolean random variables?

Problem 3

[**20 Points**] Consider the following variation on the Autoclass method for clustering. There is a set of m attributes $X_1, \ldots, X_m$, and a class Y. Assume that all the attributes and the classification are Boolean. In this variation, the model makes the independence assumption that for $j > 2$, X_j is conditionally independent of $X_1, \ldots, X_{j-2}$ given Y and X_{j-1}. The joint probability distribution for this model can be decomposed as

$$P(X_1, \ldots, X_m, Y) = P(Y)P(X_1 \mid Y) \prod_{j=2}^{m} P(X_j \mid X_{j-1}, Y)$$

You may answer the following questions by analogy with the analysis of Autoclass in the notes from Lecture 11. You do not have to provide a precise proof for any part of the question.

(a) [**4 Points**] List the parameters of the model. How many independent parameters are there, as a function of m?

(b) [**4 Points**] For n labeled instances, write maximum likelihood expressions for the parameter values, expressed in terms of sufficient statistics (which will be counts of instances with various properties.)

(c) [**4 Points**] Assume now that there is a training set of n *unlabeled* instances. Write expressions for the *expected* sufficient statistics, in terms of the training data and for some fixed set of parameter values.

(d) [**5 Points**] Combine your answers from parts (b) and (c) into an EM algorithm for this model.

(e) [**3 Points**] What are the advantages and disadvantages of using this model for clustering, compared to Autoclass? Would you expect the ordering adopted for the attributes to have a significant effect on the performance of the model as a clustering algorithm? Explain why or why not.

Problem 4

[**50 Points**] In this exercise, you will experiment with clustering algorithms on a dataset consisting of census data. You can find the following files in
`http://www.seas.harvard.edu/courses/cs181/docs/asst3.tar.gz`:

- `adults.txt:` The census data: there are 30717 instances.
- `adults-small.txt:` A subset of the census data containing only 3 attributes: age, education, and income. See `181adult.names` for explanations of how these attributes are represented.
- `181adult.names:` A description of the census data including a list of the 48 attributes (corresponding to 13 census categories). You should read this file to understand the data before you start. All feature values have been scaled to appropriate ranges. You don't need to modify the values further.
- `clust.py:` Support code in Python for loading the census data.
- `3d-scatter.p:` An example *gnuplot* script for generating a 3D scatterplot. You are not required to use this. However, you may find it helpful for part (b).

As with homeworks 1 and 2, we have included a web interface with this homework. The web interface can be run using `hw3` or `hw3.bat`. We have also created a command-line interface which may be useful to get immediate feedback from print statements or to dump the outputs to data files which can then be graphed. The code can be run with `python clust.py`. You can see the various configuration options by typing `python clust.py --help`. The script defaults to running K-means, but can be told to run the HAC algorithms with the `---run_hac` and `--hac_alg` option flags.

(a) [**20 Points**] Implement the K-means clustering algorithm. Your program should print out the means of each of the clusters. You will want to generate random initial prototype vectors by sampling from the data points.

(i) For *num-examples* $= 1000$ on the `adults` dataset (43 attributes), run the K-means algorithm with $K = 2, 3, 4, 5, 6, 7, 8, 9, 10$. For each value of K, compute the mean squared error (the mean of the squared distance of each point from its closest prototype vector). Provide a plot of this mean squared error versus K. [**Hint:** If your mean squared error results are not what you expect, remember that the initial parameters are random, so you may want to return the smallest error over several runs. You can modify the number of runs by editing the SAMPLES variable in `taskclust.py`.]

(ii) If you were to choose the best K for this data based on the the plot you generated in part (i), what value of K might you choose? Justify your choice.

(b) **[30 Points]** Implement the hierarchical agglomerative clustering (HAC) algorithm as well as the four distance metrics discussed in lecture: min, max, mean, and centroid. Your program should print out the means of each of the final clusters.

(i) For *num-examples* $= 100$, $K = 4$, run HAC on the `adults-small` data set, which contains only 3 attributes. Compare the clusters formed using the *min* distance metric against the clusters formed using the *max* distance metric. For each distance metric, submit two things: (1) a table showing the number of instances in each cluster and (2) a scatterplot of the instances in 3-dimensions, clearly indicating which instances belong to which cluster. What differences do you notice between the clusters produced using the *min* versus the *max* distance metric? Does this make sense given the definition of the metrics?

(ii) For *num-examples* $= 200$, $K = 4$, run HAC on the `adults-small` data set, which contains only 3 attributes. Compare the clusters formed using the *mean* distance metric against the clusters formed using the *centroid* distance metric. For each distance metric, submit two things: (1) a table showing the number of instances in each cluster and (2) a scatterplot of the instances in 3-dimensions, clearly indicating which instances belong to which cluster. What differences do you notice between the clusters produced using the *mean* versus the *centroid* distance metric? Does this make sense given the definition of the metrics?