Assignment 3: Classifiers, Clustering

10-701/15-781: Machine Learning (Fall 2004)

Out: Oct. 14th, 2004
Due: Oct. 28th 2004, Thursday, Start of class,

a *This assignment has four problems to test your understanding about classifiers and regression. Each of problems 1, 2, 3, and 5 are worth 15%, with problem 4 worth 40%.*

b *For the questions requiring programming, please use Matlab. You need to submit your code to the TAs (we'll provide instructions on how soon).*

c *For questions and clarifications, contact Max (questions 1 to 3) (maxim+@cs.cmu.edu) or Dave (questions 4 and 5) (dif+781@cmu.edu).*

d *Policy on collaboration:*

Homeworks will be done individually: each student must hand in their own answers. It is acceptable, however, for students to collaborate in figuring out answers and helping each other solve the problems. We will be assuming that, as participants in a graduate course, you will be taking the responsibility to make sure you personally understand the solution to any work arising from such collaboration.

d *Policy on late homework:*

Homework is worth full credit at the beginning of class on the due date. It is worth half credit for the next 48 hours. It is worth zero credit after that. You must turn in all of the 4 homeworks, even if for zero credit, in order to pass the course.

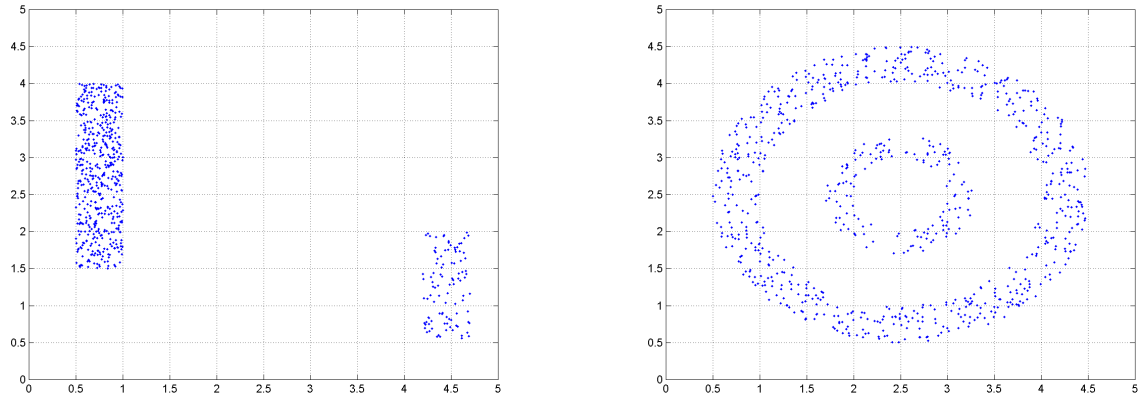
Question 1. Naive and Joint Bayes Classifiers

1.1 Suppose A and B are independent binary random variables, each having a 50% chance of being 0. Construct a boolean function $y = f(A, B)$ where A is not independent of B given y but for which a naive Bayes classifier will have a 0% error rate (assuming infinite training data). Prove that the classifier will have 0% error rate.

1.2 Suppose we have a function $y = (A \wedge B) \vee \neg(B \vee C)$, where A, B, C are again independent binary random variables, each having a 50% chance of being 0.
a) How many parameters a naive Bayes classifier needs to estimate (without counting $P(\neg x)$ as a parameter if $P(x)$ is already counted as an estimated parameter)? What will be the error rate of the naive Bayes classifier (assuming infinite training data)?
b) How many parameters a joint Bayes classifier needs to estimate? What will be the error rate of the joint Bayes classifier (assuming infinite training data)?
c) Suggest a Bayes classifier that will need to estimate less number of parameters than a joint Bayes classifier but will still have a zero error rate (assuming infinite training data). Show that the classifier has this error rate.

Question 2. Spectral Clustering I

In this problem we will analyze the operation of one of the variants of spectral clustering methods on two datasets shown in Figure 1. For each of the datasets (unless directed otherwise) please answer the following questions.



(a) (b)

Figure 1: Plots for problem 2

2.1 The first step is to build an affinity matrix. The matrix defines the degree of similarity between points.
a) Suppose we use the L2 norm to construct the following affinity matrix (let x_i denote an ith datapoint):

$$A(i,j) = A(j,i) = \begin{cases} 1 & \text{if } |x_i - x_j|_2 < \Theta \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

What θ value would you choose and why?

b) Suppose instead we use Gaussian kernel for our affinity matrix:

$$A(i,j) = exp\left(-\frac{|x_i - x_j|_2}{2\sigma^2}\right) \tag{2}$$

What σ value would you choose and why?

2.2 The second step is to compute first k dominant eigenvectors of the affinity matrix, where k is the number of clusters we want to have
a) For the dataset in Figure 1(a) and the affinity matrix defined by equation 1 is there a value of θ for which you can compute analytically eigenvalues corresponding to the first two dominant eigenvectors? If not, explain why not. If yes, compute and write these eigenvalues down.

2.3 The third step is to cluster the rows of the matrix Y into k clusters using K-means (or a similar algorithm), where Y is constructed by placing k dominant eigenvectors into columns and re-normalizing the rows (to make each row a unit vector).
a) For the dataset in Figure 1(a) and the affinity matrix defined by equation 1 write down your best guess for the coordinates of $k = 2$ cluster centers.

2.4 Finally, given the clusters on matrix Y, a point x_i is declared to be in cluster j iff the ith row of Y is in cluster j.
a) What are the final clusters you would expect to obtain for each of the datasets? Provide a rough sketch of the clusters to give an idea.
b) What are the clusters you would expect to obtain if using EM algorithm for Gaussian Mixture Models with 2 clusters? Also provide a rough sketch of the clusters.

Question 3. Spectral Clustering II

The version of spectral clustering we have studied in class made use of matrix $A = D^{-1/2}WD^{-1/2}$. W is an affinity matrix with $w_{ij} = w_{ji}$ being a non-negative distance between points x_i and x_j. D is a diagonal

matrix whose i^{th} diagonal element, d_{ii}, is the sum of W's i^{th} row. In the following you will need to prove several properties about A that are important for a good understanding of spectral clustering.

For the proofs you might find useful the following property: for any symmetric matrix B with all non-negative entries if u is an eigenvector with all positive entries, then no other independent eigenvector of B has the same eigenvalue.

3.1 Show that a vector $v_1 = [d_{11}^{\frac{1}{2}}\ d_{22}^{\frac{1}{2}}\ \ldots\ d_{nn}^{\frac{1}{2}}]^T$ is an eigenvector of A with an eigenvalue $\alpha_1 = 1$.

3.2 Prove that $\alpha_1 = 1$ is the largest eigenvalue of A.

3.3 Prove that all eigenvectors orthogonal to v_1 will have an eigenvalue stricly smaller than 1.

This property shows that if points are viewed as vertices in a Markov graph with transition probabilities proportional to distances between points (elements of W), then v_1 is the only eigenvector needed to compute the probability distribution over states matrix P^∞ (whose $(ij)^{th}$ element shows the probability of being at state j if starting at state i) after infinitely many steps.

3.4 Show that $P^\infty = D^{-\frac{1}{2}}(v_1 v_1^T)D^{\frac{1}{2}}$, where $P = D^{-1}W$ is the probability transition matrix.

Question 4. Classifiers

In this problem you will implement two common classifiers and use them to predict attribute values for a series of data. The data you will be working with is a series of records containing 4 continuous-valued attributes and one Boolean class attribute. You will be attempting to predict the Boolean attribute value based on the values of the first four.

All implementation should be done in Matlab. For each problem, we specify the prototype of the required function(s). Please follow these prototypes! It should make your life easier and keep your TA's from getting crotchety.

4.1 Create a naïve Bayes classifier with the assumption that each attribute value for a particular record is generated from a Gaussian with μ and σ determined by the class of the record. In other words, if a record is from class 1, then its first four attributes (a_1, a_2, a_3, a_4) will have their values generated from a Gaussian centered at μ_1 with diagonal covariance matrix Σ_1. You're going to use this assumption to classify a series of new records.

Here is the prototype of the matlab function you need to implement:

$$\textbf{function}\ \textit{percent\_correct} = \textbf{gaussian\_naive\_classify}(\textit{training\_data}, \textit{testing\_data}) \tag{3}$$

training_data is Gaussian-generated input data (with an arbitrary number of rows), where each row contains 4 continuous values and one Boolean value. *testing_data* is of the same format. *percent_correct* is the percent of records from *testing_data* whose classes were accurately predicted.

Report the accuracy of your classifier when using the training data given in 'ind_training_data.txt' and testing data in 'ind_testing_data.txt'.

4.2 Create a joint classifier with the assumption that each record has its attribute vector generated from a single multi-dimensional Gaussian with mean vector and covariance matrix determined by the class of the record. So, if a record is from class 1, then its attribute vector (a_1, a_2, a_3, a_4) is generated from a Gaussian centered at μ_1 with a full (non-diagonal) covariance matrix Σ_1.

Here is the prototype of the matlab function you need to implement:

$$\textbf{function}\ \textit{percent\_correct} = \textbf{gaussian\_joint\_classify}(\textit{training\_data}, \textit{testing\_data}) \tag{4}$$

again, *training_data* is input data (with an arbitrary number of rows), where each row contains 4 continuous values and one Boolean value. *testing_data* is of the same format. *percent_correct* is the percent of records from *testing_data* whose classes were accurately predicted.

Report the accuracy of your classifier when using the training data given in 'dep_training_data.txt' and testing data in 'dep_testing_data.txt'.

4.3 Now find a general algorithm for performing classification that assumes each attribute is generated independently, but each from a different arbitrary distribution based on the class of the record. Your approach should use a set of training data to optimally split each attribute's continuous space into a collection of non-overlapping intervals, with each interval having an associated class prediction.

For each attribute, you should find the number of intervals (at most 3) that provides the best class prediction over the training data. These intervals can then be used for classifying new records for testing.

[Hint: from training data we can compute $P(class = 1 \mid a_i \in X)$ for each interval X associated with attribute a_i. When given a new record, we can use the independence assumption to multiply these probabilities together and see which class is more likely overall.]

Here is the prototype of the matlab function you need to implement:

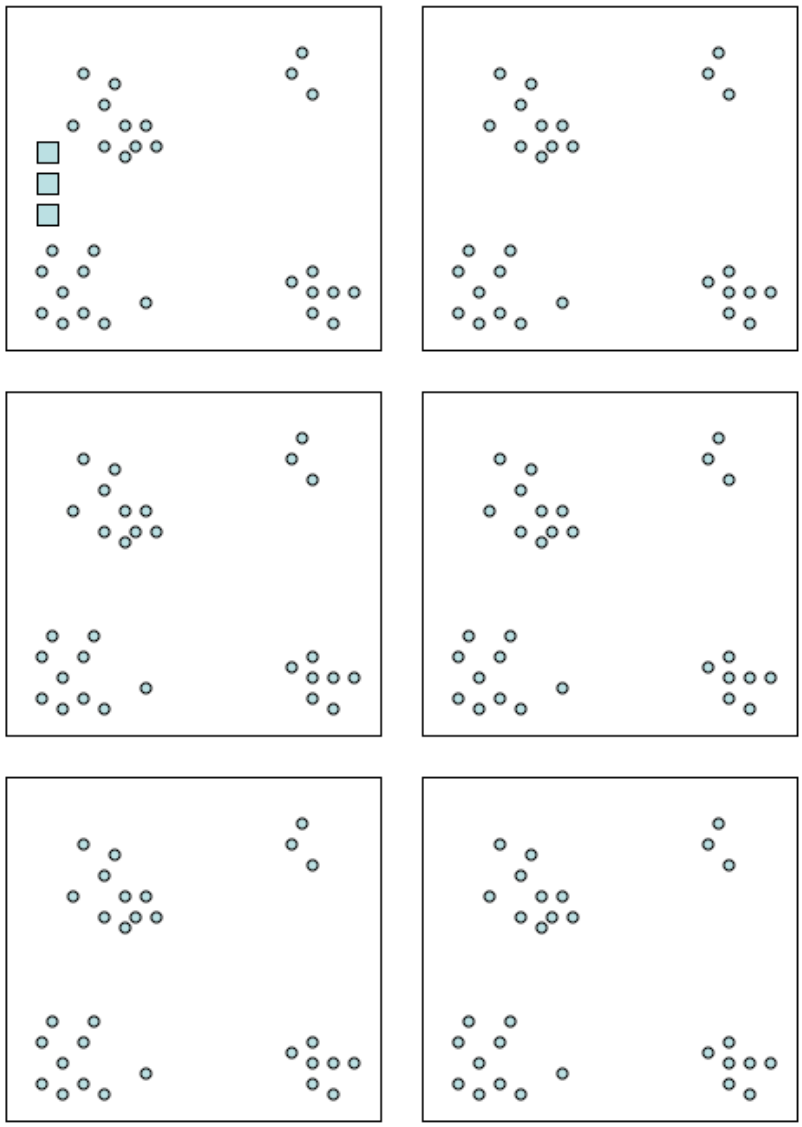
$$\textbf{function } \textit{percent\_correct} = \textbf{general\_naive\_classify}(\textit{training\_data}, \textit{testing\_data}) \tag{5}$$

again, *training_data* is input data (with an arbitrary number of rows), where each row contains 4 continuous values and one Boolean value (as in training_data.txt). *testing_data* is of the same format. *percent_correct* is the percent of records from *testing_data* whose classes were accurately predicted.

Report the accuracy of your classifier when using the training data given in 'ind_training_data.txt' and testing data in 'ind_testing_data.txt'. Also report the number of intervals used for each attribute. Why is this number not the same for each attribute, and what does this mean about the underlying Gaussian distributions used to generate each attribute's value? How does the performance of this approach compare with the naïve Gaussian classifier from 4.1? Why is it different/the same?

Question 5. K-means and Hierarchical Clustering

5.1 Run K-means manually on the following dataset. Circles are data points and squares are the initial cluster centers. Draw the cluster centers and the decision boundaries that define each cluster. If no points belong to a particular cluster, assume its center does not change. Use as many of the pictures as you need for convergence.



5.2 Using hierarchical clustering (with the minimum distance criteria), what is the maximum height of the hierarchy tree required to cluster a set of N points? What is the minimum height? Give an example of a collection of 16 points that requires (i) a hierarchy tree of maximum height, and (ii) a hierarchy tree of minimum height.

5.3 (a) Is it possible to have Gaussian Mixture Models exhibit equivalent behavior to K-means by restricting the Gaussians used? How?

(b) For what sort of data do general Gaussian Mixture Models produce much better results than K-means? Provide an example of such a dataset.