Stony Brook University
Department of Computer Science
CSE 353: Machine Learning
(Spring 2015)
Instructor: Luis E. Ortiz

Programming Homework 5: Unsupervised Learning, Clustering, Density Estimation, Mixture of Independent Gaussians, K-means; Application: Clustering for Robot Localization; EXTRA CREDIT: Clustering Emails into Spam and Ham

1 Unsupervised Robot Indoor-Localization using Generalized K-means and Gaussian Naive-Bayes Clustering

In this assignment you will explore unsupervised learning via clustering techniques to group and identify three possible room locations that a robot previously visited from noisy two-dimensional coordinate observations. The training data file is `robloc.dat`.

1.1 Clustering by Gaussian Naive-Bayes Models

Apply the *mixture-of-Gaussians (MoG)* model with the naive-Bayes assumption, i.e., *Gaussian naive-Bayes model*, to the robot-localization 2-d coordinate data set, so that the *number of features* $n = 2$, using $K = 3$ the *(Gaussian) components*.

- Run the *Expectation-Maximization (EM) algorithm* for MoG, described in the next section, on this data for $T = 20$ iterations, using the following initialization of the model parameters:

$$\pi_1^{(0)} = 0.5 \text{ and } \boldsymbol{\mu}_{\cdot|1}^{(0)} = \begin{bmatrix} \mu_{1|1}^{(0)} \\ \mu_{2|1}^{(0)} \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \end{bmatrix},$$

$$\pi_2^{(0)} = 0.3 \text{ and } \boldsymbol{\mu}_{\cdot|2}^{(0)} = \begin{bmatrix} \mu_{1|2}^{(0)} \\ \mu_{2|2}^{(0)} \end{bmatrix} = \begin{bmatrix} 2.5 \\ 1.5 \end{bmatrix},$$

$$\pi_3^{(0)} = 0.2 \text{ and } \boldsymbol{\mu}_{\cdot|3}^{(0)} = \begin{bmatrix} \mu_{1|3}^{(0)} \\ \mu_{2|3}^{(0)} \end{bmatrix} = \begin{bmatrix} 1.5 \\ 2 \end{bmatrix},$$

  and for all cluster labels $c \in \{1, \ldots, K\}$ and all features $i \in \{1, \ldots, n\}$, $\sigma_{i|c}^{(0)} = 0.5$

- Record and provide (via a table) the EM-learned estimates of the model parameters, π_c, $\boldsymbol{\mu}_{i|c}$, and $\boldsymbol{\sigma}_{i|c}^2$ for all features i and cluster labels c.

- Plot the robot-location coordinate examples in the data set, appropriately identifying each example according to its corresponding learned component, as well as the resulting means representing each Gaussian component found for each feature.

- (**EXTRA CREDIT**) Also include, for each cluster, the contour map/level sets corresponding *one* and *two standard deviations* of the Gaussian associated to that cluster.)

1.2 Learning Multi-Class Gaussian Naive-Bayes Models (*aka* Multi-Component Mixture of Independent Gaussians)

A multi-class Gaussian naive Bayes is a MoG model in which each feature i, modeled as a real-valued random variable X_i, is *conditionally independent* given its (hidden) cluster label, modeled as another discrete random variable C taking values in $\{1, \ldots, K\}$.

The *mixture-proportion parameter* π_c corresponds to the *prior* probability of a particular cluster label $c \in \{1, \ldots, K\}$ that C can take. The mixture-proportion parameters $\boldsymbol{\pi} \equiv (\pi_1, \ldots, \pi_K)$ define the *probability mass function (PMF)* p_C for C: i.e., $\mathbf{P}(C = c) = p_C(c) \equiv \pi_c \geq 0$ for all c, and $\sum_{c=1}^{K} \pi_c = 1$.

Just as for classification, for each continuous feature i, we use a Gaussian distribution $f_{X_i|C}$ as the conditional *probability density function (PDF)* of X_i given uncertain/hidden binary class label C. Hence, for a particular class label c, given corresponding *conditional mean* $\mu_{i|c}$ and individual *variance parameters* $\sigma^2_{i|c}$ for the mixture-component c, we have, for each continuous feature value x_i, the conditional *marginal* PDF

$$f_{X_i|C}\left(x_i \;\middle|\; c\,;\, \mu_{i|c}, \sigma^2_{i|c}\right) \equiv \frac{1}{\sqrt{2\pi}\sigma_{i|c}} \exp\left(-\frac{1}{2}\frac{(x_i - \mu_{i|c})^2}{\sigma^2_{i|c}}\right),$$

so that the conditional *join* PDF over *all* the continuous features (random variables) $\mathbf{X} = (X_1, \ldots, X_n)$ is

$$f_{\mathbf{X}|C}\left(\mathbf{x} \;\middle|\; c\,;\, \boldsymbol{\mu}_{\cdot|c}, \boldsymbol{\sigma}^2_{\cdot|c}\right) \equiv \prod_{i=1}^{n} f_{X_i|C}\left(x_i \;\middle|\; c\,;\, \mu_{i|c}, \sigma^2_{i|c}\right) \;,$$

for all possible attribute values $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n$ that $\mathbf{X}$ can take. To reduce notation slightly, in what follows, I will denote $f\left(x_i \;\middle|\; c\,;\, \mu_{i|c}, \sigma^2_{i|c}\right) \equiv f_{X_i|C}\left(x_i \;\middle|\; c\,;\, \mu_{i|c}, \sigma^2_{i|c}\right)$ and $f\left(\mathbf{x} \;\middle|\; c\,;\, \boldsymbol{\mu}_{\cdot|c}, \boldsymbol{\sigma}^2_{\cdot|c}\right) \equiv f_{\mathbf{X}|C}\left(\mathbf{x} \;\middle|\; c\,;\, \boldsymbol{\mu}_{\cdot|c}, \boldsymbol{\sigma}^2_{\cdot|c}\right)$ when clear from context.

Using the definitions above, the *MoG join PDF* is then

$$f_{\mathbf{X}}(\mathbf{x}\,;\, \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\sigma}^2) = \sum_{c=1}^{K} \pi_c\, f(\mathbf{x} \mid c; \boldsymbol{\mu}_c, \boldsymbol{\sigma}^2_c) = \sum_{c=1}^{K} \pi_c \prod_{i=1}^{n} f\left(x_i \;\middle|\; c\,;\, \mu_{i|c}, \sigma^2_{i|c}\right).$$

Once again, to reduce notation slightly, I will denote $f(\mathbf{x}\,;\, \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\sigma}^2) \equiv f_{\mathbf{X}}(\mathbf{x}\,;\, \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\sigma}^2)$ when clear from context.

1.2.1 The EM Algorithm for Gaussian Naive-Bayes Models

Because the labels are hidden (i.e., this is an unsupervised learning problem), we can not just count the number of examples corresponding to a specific cluster (that is exactly what we would like to find out!). Instead, we learn the parameters of the model iteratively using EM. At each round, we use the *expected counts* with respect to our current parameter estimates. In particular, at each iteration $t = 1, \ldots, T$, for each given cluster label c, denote the conditional *marginal* PDF for feature i by

$$f^{(t)}(x_i \mid c) \equiv f(x_i \mid c\,;\, \mu^{(t)}_{i|c}, [\sigma^2_{i|c}]^{(t)}),$$

and the conditional *join* PDF for input feature vector $\mathbf{x}$ by

$$f^{(t)}(\mathbf{x} \mid c) \equiv f(\mathbf{x} \mid c\,;\, \boldsymbol{\mu}^{(t)}_{\cdot|c}, [\boldsymbol{\sigma}^2_{\cdot|c}]^{(t)}) = \prod_{i=1}^{n} f^{(t)}(x_i \mid c) \;.$$

Also, denote the join PDF for input feature vector $\mathbf{x}$ at time t by

$$f^{(t)}(\mathbf{x}) \equiv f(\mathbf{x}\,;\, \pi^{(t)}, \boldsymbol{\mu}^{(t)}, [\boldsymbol{\sigma}^2]^{(t)}) = \sum_{c=1}^{K} \pi_c^{(t)} f^{(t)}(\mathbf{x} \mid c)\ .$$

Expectation step (E-step). Compute the probability/likelihood $\gamma_{l,c}^{(t)}$ that the lth example in the data set belongs to cluster c given the value of the MoG parameters at time t:

$$\gamma_{l,c}^{(t)} \equiv \mathbf{P}\left(C = c \,\middle|\, \mathbf{X} = \mathbf{x}^{(l)}\,;\, \boldsymbol{\mu}^{(t)}, [\boldsymbol{\sigma}^2]^{(t)}\right) = \frac{\pi_c^{(t)} f^{(t)}(\mathbf{x}^{(l)} \mid c)}{f^{(t)}(\mathbf{x}^{(l)})}\ .$$

Also compute the *expected* number of examples $m_c^{(t)}$ of class c given the model at iteration t:

$$m_c^{(t)} = \sum_{l=1}^{m} \gamma_{l,c}^{(t)}\ .$$

Maximization step (M-step). Reset the model parameters to the MLEs with respect to the *expected complete-data log-likelihood* using the values obtained in the E-step: that is,

$$\widehat{\pi}_c^{(t+1)} = \frac{m_c^{(t)}}{m},\ \widehat{\mu}_{i|c}^{(t+1)} = \frac{\sum_{l=1}^{m} x_i^{(l)} \gamma_{l,c}^{(t)}}{m_c^{(t)}},\ \text{and}\ [\widehat{\sigma}_{i|c}^2]^{(t+1)} = \frac{\sum_{l=1}^{m} \left(x_i^{(l)} - \widehat{\mu}_{i|c}^{(t+1)}\right)^2 \gamma_{l,c}^{(t)}}{m_c^{(t)}}.$$

When learning stops (after T rounds), assuming the final parameters at the last round are, for all cluster labels c, $\widehat{\pi}_c$, and for all attributes i, $\widehat{\mu}_{i,c}$ and $\widehat{\sigma}_{i,c}^2$, we assign each example $\mathbf{x}^{(l)}$ to the cluster

$$c_l \in \arg\max_c \left[\sum_{i=1}^{n} -\frac{1}{2}\frac{\left(x_i^{(l)} - \widehat{\mu}_{i|c}\right)^2}{\widehat{\sigma}_{i|c}^2} - \ln \widehat{\sigma}_{i|c}\right] + \ln \widehat{\pi}_c$$

(breaking ties arbitrarily).

1.3 Clustering by Generalized K-means

Apply K-means clustering to the robot-localization coordinate data set, with $K = 3$. You will implement a version which results from the EM algorithm for MoGs described above. The only difference from the algorithm above is in the value of $\gamma_{l,c}^{(t)}$, which for K-means corresponds to "hard" label assignments: that is,

$$\begin{aligned}\gamma_{l,c}^{(t)} \equiv& \mathbf{1}\left[c \in \arg\max_{c'} \mathbf{P}\left(C = c' \,\middle|\, \mathbf{X} = \mathbf{x}^{(l)}\,;\, \boldsymbol{\mu}^{(t)}, [\boldsymbol{\sigma}^2]^{(t)}\right)\right] = \mathbf{1}\left[c \in \arg\max_{c'} \frac{\pi_{c'}^{(t)} f^{(t)}(\mathbf{x} \mid c')}{f^{(t)}(\mathbf{x})}\right] \\ =& \mathbf{1}\left[c \in \arg\max_{c'} \pi_{c'}^{(t)} f^{(t)}(\mathbf{x} \mid c')\right]\end{aligned}$$

(breaking ties arbitrarily). Estimating the "expected" number of examples $m_c^{(t)}$ and model parameters, $\widehat{\pi}_c^{(t+1)}, \widehat{\mu}_{i|c}^{(t+1)}$, and $[\widehat{\sigma}_{i|c}^2]^{(t+1)}$ for all c and i, at each iteration t is the same as for the MoGs presented above, except that now we will use the new version of $\gamma_{l,c}^{(t)}$ given in the last equation.

1.4 Evaluating Generalized K-Means and MoG Clustering for Unsupervised Learning of Robot Indoor-Localization

Evaluate K-means by performing the same steps used to evaluate clustering by the MoG case (as listed earlier in the section on clustering by MoG). Compare and contrast the results.

2 EXTRA CREDIT: Hard and Soft Clustering of Spam Emails

Apply the *K-means clustering algorithm and Gaussian naive-Bayes models* to the SpamBase dataset.

Data Transformation: For numerical purposes, it is important to normalzie the data by computing, for each feature i, the empirical mean $\widehat{\mu}_i$ and variance $\widehat{\sigma_i}^2$ of the values of each feature of the *training examples*. Then every example l in the training set is transformed to

$$\widetilde{x}_i^{(l)} \leftarrow \frac{x_i^{(l)} - \widehat{\mu}_i}{\widehat{\sigma}_i} .$$

You need to record $\widehat{\mu}_i$, $\widehat{\sigma_i}^2$ for all features i you found on the *training data*, as described above, because you will need them to also transform the *test* data as, for every test example,

$$\widetilde{x}_i^{\text{test}} \leftarrow \frac{x_i^{\text{test}} - \widehat{\mu}_i}{\widehat{\sigma}_i} .$$

Initialization: For both K-means and MoGs initalize the means along each dimension by first sorting the values of each feature i, in ascendng order, and then assigning the average of the *bottom* and *top half* of the sorted values to $\mu_{i|-1}$ and $\mu_{i|1}$, respectively.

Perform the following steps:

1. First, apply both approaches to the (transformed) training data, using *fixed values for the mixture parameters π and conditional variances: $\pi_c = \frac{1}{2}$ and $\sigma^2_{i|c} = 1$, for all $c \in \{-1, +1\}$; so that the only parameters you will learn are the means.*

2. After learning the respective means, to evaluate the resulting clustering, "classifiy" the (transformed) training examples with respect to the resulting model and use the "ground-truth" output label of the training examples to compute the "training misclassification-error."

3. Similarly, use the output label of the (transformed) test data to compute the "test misclassificatin error."' *Keep in mind that obtaining a misclassification-error value $\epsilon > 0.5$ indicates that you must "reverse" the corresponding assignment of the output label of the model learned via unsupervised learning, so that the actual misclassification error of the learned model is $1 - \epsilon < 0.5$.*

4. Compare your results to those you obtained for Homework 4 on *supervised* email-spam filtering.

5. Finally, learn all the parameters of naive-Bayes MoG using the (transformed) training data by running the EM algorithm for 30 rounds. Then perform the same evaluation and comparison of training and test misclassification error described in the previous steps.

What to Turn In

You need to submit the following (electronically via Blackboard):

1. A **written report** (*in PDF*) that includes the values and plots required, and very briefly describe your experience, observations, and conclusions.

2. All your **code and executable** (as a tared-and-gziped compressed file), with instructions on how to run your program. A platform-independent executable is preferred; otherwise, also provide instructions on how to compile your program. Please use standard tools/compilers/etc., generally available in most popular platforms.

Collaboration Policy: *It is OK to discuss the homework with peers outside your own group, but each group of students must write and turn in their own report, code, etc., based on their own work. In addition, every member of the team must be able to answer any question on any aspect of the submitted report or source code.*