Homework Assignment 6: Bayesian Classifier

Due Saturday, March 25, 2023 at 11:59 pm EST

Description

In class, we studied Bayesian approaches for classification. In this assignment, we are going to use two different approaches to get an estimate of the likelihood probability $p(\boldsymbol{x}|\omega_i)$: Naïve-Bayes classifier and the maximum likelihood estimate. To classify a given testing sample/vector $\boldsymbol{x}_{test}$, we use the estimates of the likelihood probability to compute the discriminant function of each class ω_i, i.e.,

$$g_i(\boldsymbol{x}_{test}) = \ln p(\omega_i|\boldsymbol{x}_{test}) = \ln p(\boldsymbol{x}_{test}|\omega_i) + \ln p(\omega_i)$$

We assign (i.e., classify) $\boldsymbol{x}_{test}$ to the class with the highest discriminant function (which is equivalent to the class with the highest probability).

What to submit

Create a folder: `ps6_xxxx_LastName_FirstName` and add in your solutions: (xxxx = matlab or python)

`Ps6_xxxx_LastName_FirstName/`

- input/ - input images, videos or other data supplied with the problem set.
- output/ - directory containing output images and other files your code generates.
- ps6.m or ps6.py - code for completing each part, esp. function calls.
- *.m or *.py Matlab/Python function files, or any utility code
- ps6_report.pdf - a PDF file with all output images and text responses

Zip it as `ps6_xxxx_LastName_FirstName.zip`, and submit on Canvas.

Guidelines

1. Include all the required images in the report to avoid penalty.
2. Include all the textual responses, outputs and data structure values (if asked) in the report.
3. Make sure you submit the correct (and working) version of the code.
4. Include your name and ID on the report.
5. Comment your code appropriately.
6. Please avoid late submission. Late submission is penalized and not accepted after 24 hrs.
7. Plagiarism is prohibited as outlined in the syllabus and in Pitt Guidelines on Academic Integrity.

Questions

0. Data Preprocessing

For this assignment, we use the training and testing samples in 'hw4_data3.mat' (from HW4): X_train and y_train are the training examples and the corresponding class labels, respectively. Each row corresponds to one training vector. The rows of X_test are the testing vectors that you are required to predict their class, and y_test are the ground truth labels that you can use to compute the accuracy. For have 125 training samples, and 25 testing samples from 3 classes. Each feature vector is a 4x1 vector that has four feature vlaues.

a. Since we have three classes, split X_train into three subsets: one for each class' samples (i.e., X_train_1, X_train_2, and X_train_3). Each subset must have training samples that corresponds to one and only class; for example, X_train_1 should contain the samples whose label, y_train, is 1. We will use these three subsets to estimate the parameters of the likelihood function of each class, i.e., $p(\boldsymbol{x}|\omega_i),\ i = 1, 2, 3.$

Output: (`textual response`):
- Size of X_train_1, X_train_3, and X_train_3.

1. **Naïve-Bayes classifier**

The Naïve-Bayes classifier (discussed in section 2.5.7; see attached handout) is one of the most simple and robust classifiers. In this problem, you are asked to implement a Naïve-Bayes classifier and test its performance on real problem. The main assumption of a Naïve-Bayes classifier is that all features are statistically independent. Therefore,

$$p(\boldsymbol{x}|\omega_i) = \prod_{j=1}^{n} p\left(x_j|\omega_i\right) \quad (1)$$

$$\ln\left(p(\boldsymbol{x}|\omega_i)\right) = \sum_{j=1}^{n} \ln\left(p\left(x_j|\omega_i\right)\right)$$

where $\boldsymbol{x} = \left[x_1, \dots, x_j, \dots, x_n\right]^T$ is the feature vector with n features, x_j is the j^{th} feature, and ω_i is the i^{th} class. For our data set we have four features, i.e., $n = 4$, we have three classes $\omega_i, i = 1, 2, and\ 3.$

In this part, besides the assumption of feature independence, we are going to assume that each feature x_j follow a Gaussian distribution. Therefore, we need to, separately, compute the mean and standard deviation for each feature given a particular class ω_i. Thus, in this problem, we are going to assume that each feature (x_j ; $j = 1\ to\ n$) from vectors belonging to class ω_i has a Gaussian distribution with mean $\mu_{j|i}$ and standard deviation $\sigma_{j|k}$.

a. For each class ω_i, calculate the mean and the standard deviation of each feature (i.e., column of the corresponding class training matrix X_train_i, $i = 1, 2, 3$). After finishing this step, you should now have 12 values for the mean (one for each feature per each class data) and 12 values for the standard deviation. Therefore, you can calculate $p\left(x_j|\omega_1\right), p(x_j|\omega_2)$

and $p(x_j|\omega_3)$ using the normal distribution equation for every feature $x_j; j = 1, 2, \dots, 4$. For example:

$$p(x_j|\omega_1) = \frac{1}{\sqrt{2\pi}\ \sigma_{j|1}} \exp\left(-\frac{(x_j - \mu_{j|1})^2}{2\,\sigma_{j|1}^2}\right)$$

$$where\ \mu_{j|1}\ (\sigma_{j|1}\ ) \text{ is the mean (standard deviation) of the j}^{\text{th}} \text{ feature,}$$
computed using the training samples that belong to class 1

Output: (`textual response`):

- A table that list the values of mean and standard deviation for each feature and each class.

b. Assume that the three classes are equally probable, i.e., $p(\omega_i) = \frac{1}{3}$ for $i = 1, 2, \text{and } 3$. For each entry (row) in the testing set, make a prediction:

I. For each feature j, calculate $p(x_j|\omega_1), p(x_j|\omega_2)$ and $p(x_j|\omega_3)$.

II. Use equation (1) to calculate $\ln(p(\boldsymbol{x}|\omega_i))$ for $i = 1, 2, \text{and } 3$.

III. Compute the log of the posterior probabilities for classes 1, 2 and 3:
$$\ln(p(\omega_i|\boldsymbol{x})) \approx \ln(p(\boldsymbol{x}|\omega_i)) + \ln(p(\omega_i)) \quad \text{for } i = 1, 2, \text{and } 3.$$

IV. Assign/classify this testing entry to the class that gives the highest posterior probability $\ln(p(\omega_i|\boldsymbol{x}))$. For example, if $\ln(p(\omega_3|\boldsymbol{x}))$ is higher than $\ln(p(\omega_1|\boldsymbol{x}))$ and $\ln(p(\omega_2|\boldsymbol{x}))$, we classify $\boldsymbol{x}$ as class 3.

V. Compare your prediction to the actual class value found in Y_test.

Output: (`textual response`):

- Report the accuracy of your classifier

2. Max likelihood and Discriminant function for classification

Unlike the Naïve approach that assume statistical independence of each feature. We can use maximum likelihood estimation (MLE) to get an estimate of the data covariance matrix without assuming anything about the independence of the individual features. For this part, we use the same dataset, and we assume that the data follow a multidimensional Gaussian distribution. Given N feature vectors, the MLE estimate of the mean (derived in class) and the covariance matrix is as follows:

$$\mu = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{x}^{(i)},$$

$$\Sigma = \frac{1}{N-1}\sum_{i=1}^{N} (\boldsymbol{x}^{(i)} - \boldsymbol{\mu})(\boldsymbol{x}^{(i)} - \boldsymbol{\mu})^{\boldsymbol{T}},$$

where $\boldsymbol{x}^{(k)}\ is\ the\ k^{th}\ feature\ vector.$

We will use built-in functions to get the mean and covariance matrix of the likelihood function of each class, i.e., $p(\boldsymbol{x}|\omega_i)$. Then, to classify a feature vector we compute the discriminant function for each class.

a. Get an estimate of the covariance matrix $\Sigma_i, i = 1, 2, 3$ for each of the three classes, using the training samples for that class. For example, the covariance matrix of class 3 can be obtained using this MATLAB/Python instruction: `Sigma_3 = cov(X_train_3).`

   Output:
   - Size of each covariance matrix
   - A snapshot of your covariance matrices.

b. The mean vector for class i is simply $\mu_i = [\mu_{1|i}, \mu_{2|i}, \cdots, \mu_{4|i}]^T$, where each element is the average of the corresponding feature given data samples from the i-th class. Use the values of the mean you obtained in part 1.a or the MATLAB/Python `mean` function; compute the vectors μ_1, μ_2 and μ_3; the mean vector for class 1, class 2, and class 3, respectively.

   Output:
   - Size of each mean vector
   - A snapshot of your vectors.

c. Assume that the three classes are equally probable, i.e., $p(\omega_i) = \frac{1}{3}$ for $i = 1, 2, \text{and } 3$. For each entry (row) in the testing set, $\boldsymbol{x}$, make a prediction:

   i. Compute $g_1(\boldsymbol{x}), g_2(\boldsymbol{x})$ and $g_3(\boldsymbol{x})$; the discriminant function for each class computed on that testing feature vector $\boldsymbol{x}$. Use the equation in Lecture 14, slide 24 (page 12).
   ii. Assign this testing entry to the class with the maximum value of the discriminant function. For example, if $g_2(\boldsymbol{x})$ is higher than $g_1(\boldsymbol{x})$ and $g_3(\boldsymbol{x})$, we classify $\boldsymbol{x}$ as class 2.
   iii. Compare your prediction to the actual class value found in Y_test, and compute your classifier's accuracy.

   Output (`textual response`):
   - Report the accuracy of your classifier.
   - Compare between the naïve classifier and the MLE based classifier.