Autonomous Agents

Assignment 2: Single Agent Learning 5 October, 2012

By:
Paris Mavromoustakos
Georgios Methenitis
Marios Tzakris

Introduction

In this lab assignment we have built a probabilistic classifier for one particular purpose: to point out how the complexity of learning is modified according to the number of variables observed. To observe the difference we implemented two different models, a Bayesian classifier with normally distributed datapoints, and a Bayesian classifier with datapoints deriving from a mixture of Gaussians (MOG). The two models are represented graphically below.

A mixture distribution is the probability distribution of a random variable whose values can be interpreted as being derived from an underlying set of random variables. In the above figure, it is shown that a **latent variable** z underlies between class C and datapoint x, thus latent variable z controls which datapoints belong to which mixture element.

Exercise 1

First, we loaded the given data files `banana.mat`, and, `spiral.mat`, each one of them contains two-dimensional data points, represent two different classes. The training set, which we used in order to train our classifier consists of 75% of data points from class A, appended to 75% of data points from class B, while the test set contains the remaining 25% of both classes A and B. Figure 1, shows the 2-D representation of the two classes of data-points.

Looking into these data-points, we can figure out that the distribution of the data in both cases is following a "banana" and a spiral shape respectively. Therefore, a single 2-D Gaussian distribution could not be able to describe the class conditional probabilities well. For this reason, we do not expect the model to perform accurately. In order to perform the training, we need the prior probabilities, the mean μ_k of each class and the covariance matrix Σ_k for each class $k \in \{A, B\}$. We set the prior probabilities 0.5 as the number of the data for class A and B is the equal. Next, we can directly extract the mean and and the covariance for each class directly from the training data. The pseudocode for that is given below:

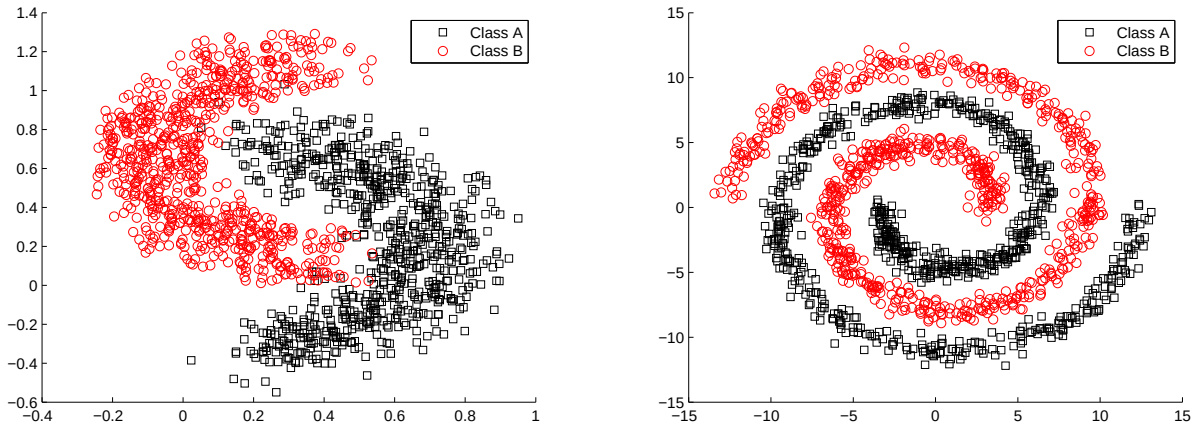

Figure 1: Depiction of the 2-D data-points of the two given classes for the `banana.mat` (left) and `spiral.mat` (right).

```
[A, B] = load( data_points )
TrainingSetA = A(1:length(A)*3/4,:)
TrainingSetB = B(1:length(B)*3/4,:)
TrainingSet = [TrainingSetA; TrainingSetB]

prior_probability_A = size(A,1)/(size(A,1)+size(B,1))
prior_probability_B = size(B,1)/(size(A,1)+size(B,1))

for i=1:size(TrainingSetA,1)
   sumAx = sumAx + TrainingSetA(i,1);
   sumAy = sumAy + TrainingSetA(i,2);
   sumBx = sumBx + TrainingSetB(i,1);
   sumBy = sumBy + TrainingSetB(i,2);
end

mean_A = [sumAx sumAy]/size(TrainingSetA,1);
mean_B = [sumBx sumBy]/size(TrainingSetA,1);

covariance_A = cov(TrainingSetA);
covariance_B = cov(TrainingSetB);

pA = mvnpdf(TestSet,mean_A,covariance_A);
pB = mvnpdf(TestSet,mean_B,covariance_B);

PostA = (pA .* prior_probability_A) ./ (pA .* pr_classA + pB .* pr_classB);
PostB = (pB .* prior_probability_B) ./ (pA .* pr_classA + pB .* pr_classB);
```

After the classification process we got the following confusion matrices and error rates for the our classifier:

Banana set

$$Confusion\ Matrix = \begin{bmatrix} 226 & 24 \\ 26 & 224 \end{bmatrix},$$
$$error\ rate = 0.1$$

Spiral set

$$Confusion\ Matrix = \begin{bmatrix} 174 & 76 \\ 89 & 161 \end{bmatrix},$$
$$error\ rate = 0.33$$

As we said before, the way that we tried to perform classification in this type of data, did not perform well, mainly because, the shape of the data, and the fact that, data-points from the one class are located inside the other class, make it impossible for a simple 2-D Gaussian distribution to represent them.

Exercise 2

Exercise 3

Conclusion