Machine Learning

Lab 7 - Naive Bayes
Spring 2025

Introduction

In this lab you will implement a probablistic classifier, namely *naive bayes* for multi-class classification.

You may **not** use any functions from a ML library in your code. And as always your code should work on any dataset that has the same general form as the provided one.

Grading

- +1pt Can run script properly.
- +1pt Parses dataset properly.
- +1pt Generates requested info for Part 1.
- +1pt Better than random results for Part 1
- +2pt Good results for Part 1 ($Accuracy \approx 80\%$)
- +1pt Generates requested info for Part 2.
- +1pt Better than random results for Part 2
- +2pt Good results for Part 2 ($Accuracy \approx 75\%$)

Datasets

Cartiotocography Dataset (CTG.csv) Download the file CTG.csv from Bblearn. This file contains 2126 instances of 21 feature pertaining to information obtained from Cardiotocography tests. Our task is to determine the fetal state class code given an observation. This code can be one of the 3 values and pertains to the LAST column of the dataset. The second to last column of the dataset can also be used for classification but for our purposes DISCARD it. That is, discard the column named **CLASS**, and use the column named **NSP** as our target column.

You can read more about the dataset here:

http://archive.ics.uci.edu/ml/datasets/Cardiotocography

Yale Faces Datasaet This dataset consists of 154 images (each of which is 243x320 pixels) taken from 14 people at 11 different viewing conditions (for our purposes, the first person was removed from the official dataset so person ID=2 is the first person).

The filename of each images encode class information:

$$\text{subject}< ID >.< condition >$$

Data obtained from: http://cvc.cs.yale.edu/cvc/projects/yalefaces/yalefaces.html

1 Naive Bayes Classifier

Let's train and test a *Naive Bayes Classifier* to classifiy the fetal state from the Cartiotocography dataset.

Write a script that:

1. Reads in the data.
2. Shuffles the observations
3. Selects the first 2/3 (round up) of the data for training and the remaining for validation.
4. Pre-processes the data. Although technically some of the columns are discerete valued, let's treat them all as continuous and convert them to binary ones using the **mean** of that feature, as computed from the training data.
5. You can now using the training dataset to compute:
   (a) Class priors
   (b) Naive probabilities, $P(x_i|y)$ for each feature of each class
6. Given that information, you can now classifies each validation sample.

Your script should output the following:

1. The class priors.
2. The validation accuracy of your system.
3. Your confusion matrix.

2 Additional Dataset

Now let's see how your systems work on another multi-class dataset, the Yale Faces dataset!

As with prior labs, read in all the images, resize them to be 40×40 images (use nearest neighbors so no interpolation occurs), then reshape into 1×1600 feature vectors, such that in the end we have a 154×1600 observable data matrix. As far as pre-processing the features, as long as you used nearest neighbors for resizing your images, all the features will be discrete $\in [0, 255]$. **You can decide how/if you want to convert these to other enumerations (say, binary, or just 8 values, etc..)**.

We'll again be classifying the subject, so again you should divide your training and validation according to subject (2/3 of each subject for training, 1/3 for validating).

Your script should output the following:

1. The class priors.
2. The validation accuracy of your system.
3. Your confusion matrix.