

SGN-41007 Pattern Recognition and Machine Learning

Exercise Set 4: February 1–February 3, 2017

Exercises consist of both pen&paper and computer assignments. Pen&paper questions are solved at home before exercises, while computer assignments are solved during exercise hours. The computer assignments are marked by text `python` and Pen&paper questions by text `pen&paper`

1. `pen&paper` *Design an LDA classifier manually.*

A dataset consists of two classes, whose distributions are assumed Gaussian, and whose sample covariances and means are the following:

$$\begin{aligned}\mu_1 &= \begin{pmatrix} -1 \\ 1 \end{pmatrix} & \mu_2 &= \begin{pmatrix} -6 \\ 2 \end{pmatrix} \\ C_1 &= \begin{pmatrix} 2 & 0.1 \\ 0.1 & 0.2 \end{pmatrix} & C_2 &= \begin{pmatrix} 3 & -2 \\ -2 & 1 \end{pmatrix}\end{aligned}$$

Calculate the projection vector w . In order to be fully manual, invert the 2×2 matrix using the rule

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}^{-1} = \frac{1}{ad - bc} \begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$$

2. `pen&paper` *Compute the threshold and classify.*

The LDA decision rule requires a threshold T :

- Decide class = 1 if $w^T x \geq T$.
- Decide class = 0 if $w^T x < T$.

Compute T by setting it at the center of projected class means, μ_1 and μ_2 .

Which class will be predicted for sample $x = (1, 2)$?

3. `pen&paper` *Compute the threshold more properly and classify.*

The previous approach to defining the threshold T did not take into account the fact that the two classes have different spreads, and the threshold should probably not be exactly at the center.

A more appropriate approach would thus compute the projection of the multivariate Gaussians and set the threshold accordingly. The projected Gaussians are univariate normal: $\mathcal{N}(w^T \mu_1, w^T C_1 w)$ and $\mathcal{N}(w^T \mu_2, w^T C_2 w)$. Formulate the classification problem as a likelihood ratio test and choose the threshold based on that.

Which class will be predicted for sample $x = (1, 2)$?

4. **python** *Extract Local Binary Pattern features for classification.*

In this exercise we will extract image features for categorization of traffic signs. Download the following file:

http://www.cs.tut.fi/courses/SGN-41007/GTSRB_subset.zip

It has two folders each containing 100 images from the German Traffic Sign Recognition Benchmark (GTSRB); a competition organized in IJCNN-2011 conference.

Load all images from the two folders and compute their Local Binary Pattern features using `skimage.feature.local_binary_pattern`.¹ The function returns an image with same size as the original, so you will also have to compute the histogram with `numpy.histogram`. Note that this is a similar task to exercise 4 last week. The result should be a feature matrix `X` and label vector `y`.

5. **python** *Train classifiers for the GTSRB task.*

Create a list of three classifiers with their default parameters:

- `sklearn.neighbors.KNeighborsClassifier`
- `sklearn lda.LDA`
- `sklearn.svm.SVC`

Split your data into two parts—80% for training and 20% for testing—using `sklearn.cross_validation.train_test_split`.

Train each classifier in a for loop and assess the accuracy in the test set using `sklearn.metrics.accuracy_score`. Which one is the best?

¹See http://scikit-image.org/docs/dev/auto_examples/plot_local_binary_pattern.html