MACHINE LEARNING FROM DATA

Lab Session 4 – K-Nearest Neighbors and Parzen windows

1. Goal

The goal of this session is to

- Use and compare two non-parametric classifiers, k-nearest neighbors and Parzen windows
- Test these methods on an image dataset of handwritten digits
- Implement a simple cross-validation strategy for hyperparameter selection
- Learn to use Python pipelines

2. Instructions

- Download and uncompress the file **Mlearn_Lab4.zip**
- Answer the questions in the document **Mlearn_Lab4_report_surname.docx,** save and convert to pdf
- Write the new code in the same Colab Notebook **Mlearn_lab4_knn_parzen.ipynb**

3. Introduction and previous study

Read the slides corresponding to lecture 4: non parametric classifiers.

In this session we will use the Zip-code dataset. It is an example of classification problem where the number of samples is relatively small compared to the dimensionality of the feature space and the number of classes.

We will use two parametric models: k-nearest neighbors, where we will have to choose an optimal value for the parameter k, and Parzen windows, where we will have to select an optimal value for the parameter h.

4. Zip-code dataset

4.1. Characteristics of the dataset

The Zip code dataset can be found in this repository:
https://web.stanford.edu/~hastie/ElemStatLearn/data.html

Each element in the dataset is a vector of d=256 features corresponding to the intensity values of a 16x16 image of a handwritten digit. Images are vectorized row by row. There are 10 classes corresponding to the 10 digits (0, …, 9). The dataset is already split into training and test subsets. The training set contains 7291 samples and the test set contains 2007 samples.

This is the information provided by the authors:

```
Normalized handwritten digits, automatically scanned from envelopes by the
U.S. Postal Service. The original scanned digits are binary and of different
sizes and orientations; the images here have been deslanted and size
normalized, resulting in 16 x 16 gray scale images (Le Cun et al., 1990).

The data are in two zipped files, and each line consists of the digit id (0-
9) followed by the 256 grayscale values.

There are 7291 training observations and 2007 test observations, distributed
as follows:
```

```
         0    1   2   3   4   5   6   7   8   9 Total
Train 1194 1005 731 658 652 556 664 645 542 644 7291
 Test  359  264 198 166 200 160 170 147 166 177 2007

or as proportions:
         0    1   2    3    4    5    6    7    8    9
Train 0.16 0.14 0.1 0.09 0.09 0.08 0.09 0.09 0.07 0.09
 Test 0.18 0.13 0.1 0.08 0.10 0.08 0.08 0.07 0.08 0.09

Alternatively, the training data are available as separate files per digit
(and hence without the digit identifier in each row)

The test set is notoriously "difficult", and a 2.5% error rate is excellent.
These data were kindly made available by the neural network group at AT&T
research labs (thanks to Yann Le Cunn).
```

First, we will work with data in the original format (d=256). Then we will apply PCA and MDA to reduce the dimensionality of the feature space, working with vectors of size d'=64 (for PCA) and d'=9 (for PCA and MDA).

Next, we will apply a simple validation strategy for selecting the most appropriate hyperparameter value (that is k for kNN and h for Parzen windows).

4.2. Classification using kNN

Open Colab Notebook **Mlearn_lab3_knn_parzen.ipynb**

Read the notebook identifying the different sections in the code.

Run the code for loading the training and test dataset splits.

Run the code with the option 'no reduction' to keep vectors with all 256 features.

Answer the following questions:

Q1. Copy the results obtained with kNN on the train and test sets, and discuss the results. What is the value of *k* (default)? Analyze the confusion matrices and identify the two most challenging classes.

Q2. Run again the code, using PCA to reduce the dimensionality of the feature space, selecting d'=64 features. Observe the eigenvectors and the images reconstructed using only the first d' eigenvectors (those with the highest eigenvalues). Discuss. Copy train and test results. Discuss the results and compare with the previous case (no PCA).

Q3. Repeat the previous analysis using PCA with d'=9 features, and MDA with d'=9 features. Discuss which method is the best for image reconstruction and which one is preferable for classification.

4.3. kNN with cross validation

In this section we will use MDA with d'=9 features.
However, instead of using an arbitrary value of *k* in kNN, we will use GridsSearchCV validation with a single stratified split in order to select the optimal value of this parameter.

Q4. Find the optimal value of *k* on the training set. Use at least 10 values for *k*. Plot the train and validation errors as a function of *k*. Use the optimal value of *k* to compute the error on the test set.

4.4. Parzen windows with cross validation

Now we will use Parzen windows to classify the zip-code dataset.

You will have to add the code for this part, using the ParzenClassifier class provided in the notebook Again, reduce dimensionality to d'=9 features with MDA.

Q5. Use the same GridSearchCV strategy (with a single stratified split) to select the best parameter *h*. Try different values of *h* (for example 1, 10, 20, 30…. try more values). Plot the train and validation errors as a function of *h*. Use the optimal value of *h* to compute the error and other metrics on the test set.