EE-613
Machine Learning for Engineers
Lab session – Boosting

François Fleuret

November 9, 2019

Introduction

The objective of this practical session is to use Boosting to classify cancerous cells, and understand how Boosting can be extended to another loss.

Files

You have to download

```
https://fleuret.org/ee613/files/lab-boost.zip
```

and unzip it. It will create a directory `lab-boost`, in which you can execute `boost.py` which should print

```
train (426, 30) (426,)
test (143, 30) (143,)
```

that correspond to the shapes of the feature and target tensors for training and test respectively.

The data-set is the Wisconsin Breast Cancer data set. Each one of the 569 samples corresponds to a cell. Features are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe characteristics of the cell nuclei present in the image. The class corresponds to malignant (1) or benign (0).

The 30 real-valued features for each cell correspond to quantities related to its nuclei morphology, such as radius, texture, perimeter, area, smoothness, etc.

Pre-defined class and functions

`utils.py` contains several pre-defined utility functions and class:

- `load_wdbc_data()` loads the data and returns the feature and sample Boolean class tensors
- `errstr(errors)` expect a vector of values and returns a string with mean value and its standard deviation between parenthesis
- `nb_errors(target, output)` expect a Boolean tensor and a real-valued tensor of same dimension and returns the classification error rate.

- `split_data(nb_for_train, input, target)` select at random `nb_for_train` samples from the provided full set, where `input` is a continuous-valued tensor of shape $n \times d$ and `target` a Boolean tensor of shape n. It returns four tensors corresponding respectively to the features and classes for train, and features and classes for test.
- `BoostedClassifier` is a class with two main methods:
  - `train(input, target, nb_stumps)` expects a real-valued feature tensor of shape $n \times d$, a Boolean class tensor of shape n, and a number of stumps. It will train the model accordingly with Adaboost.
  - `forward(input)` expects a feature tensor and return a continuous-valued vector of strong-classifier responses

Question 1

In `boost.py` write the necessary code to train a Boosted classifier with 25 stumps on 75% of the samples of the Wisconsin Breast Cancer, and compute its error rate. Repeat this 100 times and prints the average number of errors and standard deviation.

Question 2

Compute the same performance when in each run you select 5 features at random.

Question 3

Write a function

```
def fisher_scores(input, target):
```

that returns a tensor with the Fisher scores

$$S_n = \frac{(\mathbb{E}(X_n \mid Y = 1) - \mathbb{E}(X_n \mid Y = 0))^2}{\mathbb{V}(X_n \mid Y = 1) + \mathbb{V}(X_n \mid Y = 0)}$$

Compute the Boosting performance as in previous questions, using only the 5 features with highest Fisher scores.

Question 4

Given a training set

$$(x_n, y_n) \in \mathbb{R}^D \times \{-1, 1\}, \ n = 1, \ldots, N,$$

we consider the loss

$$\mathscr{L}(f) = \sum_{n=1}^{N} \max(0, -f(x_n)y_n).$$

What would be the sample boosting weights to select a weak learner

$$h : \mathbb{R}^D \to \{-1, 1\}$$

with a weighted error as far as possible from 1/2?