EE-613: Machine Learning for Engineers
Practical session 11
(Feature selection)

1 Introduction

The objective of this session is to look at three different ways of selecting features: (1) randomly, (2) with a wrapper, and (3) with a filter.

We re-use the same classes as in the practical session on decision trees and the same data-set (MNIST), here with a binary labeling (class "1" vs. the rest). Since we are using decision trees, instead of *features* we will select the *questions* used to train a decision tree, which plays the role of features in that context.

2 Download and test

If you type

```
wget http://www.idiap.ch/~fleuret/files/EE613/EE613-pw14.tgz
tar zxvf EE613-pw14.tgz
cd EE613/pw14
wget http://www.idiap.ch/~fleuret/files/EE613/mnist.tgz
tar zxvf mnist.tgz
./do.sh demo
```

you should obtain after a few seconds a printout ending with

```
Training the decision tree ... done.
**********************************************************************
Using 784 questions.
Train error 0 %
Test error  0.83 %
**********************************************************************
```

3 Programming

3.1 Testing code

You can compile the source code with `make -k`.

It is suggested to implement each question in the corresponding function

```
void computation_question1(VignetteSet *train_image_set,
                           VignetteSet *test_image_set) {
  ...
}
```

and to test it by calling the `./do.sh question1`, `./do.sh question2`, etc.

3.2 Classes

A **QuestionSet** is a virtual object standing for a set of Boolean questions used in the decision tree.

```
class QuestionSet {
public:
  virtual int nb_questions() = 0;
  virtual bool response(int n_question,
                        VignetteSet *vs, int n_vignette) = 0;
};
```

The **PixelQuestionSet** is a set of $28 \times 28 = 784$ questions, each testing if a certain pixel of the vignette is greater than 128.

The class **SubQuestionSet** provides a simple mean to use a subset of questions. Its constructor takes as parameter a **QuestionSet** and an array indicating which features should be used.

4 Questions

The `compute_demo` function provides the code to compute and print the train and test errors of a decision tree trained using all the questions from the **PixelQuestionSet** on the MNIST data-set.

Question 1: Random selection of features

As in the demo, use a PixelQuestionSet and the MNIST data-set, but select 10 questions at random, and print the training and test errors.

They should be $\simeq$10-12%. A sanity check is that if you select as many question as there are in the original set (784), you obtain error rates similar to the ones of the demo.

Help: Use the **SubQuestionSet** class to extract questions from the **PixelQuestionSet**, and the function `tag_subset` from `misc.h` to select questions at random.

Question 2: Wrapper

Repeat the same process as in Question 1 one hundred times, and keep the lowest training error, and the test error associated to it.

They should be $\simeq$7-8%.

Question 3: Filter using mutual information

Do as in Question 1, but select the questions based on their mutual information with the label to predict.

The error rates should be 4-5%.

Help: You can use the function `mutual_information(int nb, int *x, int *y)` from `misc.h` to compute the mutual information between the value in `x` and the value in `y`, estimated on the provided `nb` samples, and you can use `tag_the_top_ones` to tag the best questions before calling the constructor of **SubQuestionSet**.