521289S Machine Learning
Spring 2016

Programming exercise: Classification

1. Description

This task introduces the student to the **classification** using MATLAB.

The work is to be done solo.

The result will form one-third of the course grade.

Deadline: 8.4.2016 16.00

2. Task

Your task is to implement and test a classifier on the given data using MATLAB. **In your solution, you should use more than one method trying to find the best possible solution for the case! This means that your solution has to consist of several steps and each one of these consist at least one method (for example one step for preprocessing, one for feature extraction and one for classification).** [Using more than one method does **not** mean that you make several solutions (each one consisting of one method) and compare them with each other.]

The learning data used in this task are publicly available via the "Data"-link on the course web-page (https://noppa.oulu.fi/noppa/kurssi/521289s/harjoitustyot). In addition, download the "Template" found on the same page. **Rename** it to "*classify.m*".

In the learning data (given in the matrix *trainingData*), there are 5704 samples with 8 features. The correct class (i.e. the training target) is given by the value 0, 1 or 2 in the vector *class_trainingData*.

You must use the downloaded m-file template as a basis of your code, because your code has to be compatible with our testing interface. Of course, you are free to experiment with MATLAB, but the tested code must follow the template.

You must include all the code within the single m-file based on the template. To achieve this, more complicated solutions should be using nested functions and/or subfunctions as necessary. If you are unfamiliar with these terms, please refer to MATLAB help file and take a look at the template. Hint: You may either work only within the template m-file or you can just as well first implement several functions in as many files, and before submission collect and test them in the template.

The template defines the required interface consisting of a nick name function (*getNickName*), a training subfunction (*trainClassifier*) and a classifier evaluation subfunction (*evaluateClassifier*). You must implement the code for these functions. The number and names of the input and output variables of these functions are fixed and should not be altered in any way. Naturally, this applies also to the names of the functions themselves. These function templates expect the input data to be in

the format of the downloaded training data. The output of the classifier evaluation function is a vector similar to the correct class label in the training data. There is one exception to the do-not-modify rule: you are free to choose the data format and contents of the output (*parameters*) of the training subfunction. This data structure is used to carry the parameters learned from the training data to the classifier, and its contents will depend heavily on the chosen classifier. For more details, please refer to the template.

Please note that due to the data size, especially iterative algorithms may take time. During development, you might choose to select a smaller sample to speed up the process. However, please take care that the final code can be run within the set time limit (1 hour) using all the training data supplied and with a test data set of unknown and potentially much larger size. To speed up your code, please consult MATLAB help on "Techniques for Improving Performance", especially "vectorization". In addition, please implement suitable restrictions on e.g. iteration and search tree traversal if necessary.

Take care not to over-train the given data. After being first trained on the publicly available training data set, your solution will be validated at the test server against a different private data subset that has been withheld. Hence, your code must work with differing amounts of data. Therefore, do not hard-code sample sizes in your solution.

The performance of your code is evaluated by submitting the code to our test server using the link found on the web-page of the course. You will need an account name and password for uploading. You can get them by sending an email to the assistant. You also have to have a nickname which you can freely choose by yourselves. The nickname is used on the ranking page to identify your results among others. Use only one nickname throughout the course!

Before you submit your code to the test server, make sure that your code is properly working in your own machine. Submit only error-free code! Remember that your code has to accept also differing amounts of data. It also must perform its tasks in a reasonable time because there can be several students testing their codes at the same time. One hour is a time limit.

As a result from submitting you will get an email containing your solution's performance reading (accuracy as a per cent). If there are many students testing their solutions at the same time, it can take a while before you get the email, so be patient! If you have not got the email after one day, contact the assistant. **After getting the result you can choose to make your solution better and submit a new solution as many times as you wish**. Remember to use the same nickname every time. On the course web-page there is a "Ranking list", which shows the best performance of the each nickname's tested codes. You will get an email containing performance reading after every submission, but only your nickname's best performance reading is shown in the "Ranking list".

When you are satisfied with your results and classification accuracy given by our testing server to your code is at least 67%, deliver your solution by email to the assistant to complete the assignment! Please refer to the required deliverables below!

3. Deliverables

Before delivering your solution to the assistant, test it by using the online testing procedure explained above!

Create a MATLAB m-file based on the template. Remember to **clean** and properly **comment** your code. **Write** a short (~1-page) report of your work. Remember to include all the computed values to it, especially your solution's classification accuracy 1) from your own testing and 2) from our testing server to the report. In the report, **explain** what methods you have chosen and why you have chosen those methods. If you have used a method not included in the lectures of the course, you must also explain the method more thoroughly in the report. **Attach** the **code** as an appendix to the report. Remember to also include your student number and the nickname you used to the report.

Email the report and completed code to the assistant. **The assistant will inform you whether your programming exercise is accepted or if it still needs more work.**

4. Grading

The work will be graded according to its performance, and the result will form one-third of the course grade!

Your solution must achieve

1. classification accuracy > 67 % from the course testing server and
2. acceptance from the assistant

before it is graded.

Grading table

Classification accuracy [%] as reported by the server	**Grade**
85.00000 -	5
82.00000 - 84.99999	4
78.00000 – 81.99999	3
73.00000 – 77.99999	2
67.00000 – 72.99999	1
below 67	Fail

Deadline: Programming exercise must be finished at latest **8.4.2016**!

5. Hints and tips

Matlab exercises (https://noppa.oulu.fi/noppa/kurssi/521289s/matlab_exercises) !!!

The template file, exercises and course material should provide you with plenty of help for completing this assignment successfully. If you encounter issues with the testing server or believe you have found an error in these instructions you can contact the assistant by e-mail by writing to ilkka.juuso@ee.oulu.fi