Machine Learning COIY065H7
2018-2019
Coursework description, guidelines and marking scheme

This coursework is for MSc students only

1. Introduction

This assignment is an integral part of this module and contributes 20% to the overall mark.

The aim is to specify a machine learning solution to a classification problem. You have to explain how your method works and adjust its free parameters. You have to discuss your experimental set up and explain your experiments. Lastly, you have to evaluate how the method solves the problem, and discuss your results

By doing this coursework you will get experience with running, adapting, and evaluating machine learning methods on real data. You will need to write, reuse or change code, run it on some data, make some figures, read a few background papers, present your results, and write a report describing the problem you tackled, the machine learning algorithm you used and the results you obtained.

You have to choose one of the following problems:

- **Yeast dataset**: problem information and data are available on the UC Irvine Machine Learning Repository- http://archive.ics.uci.edu/ml/index.php
- **E.coli dataset**: problem information and data are available on the UC Irvine Machine Learning Repository- http://archive.ics.uci.edu/ml/index.php
- **Detection of malignant regions in colonoscopy video frames**: this dataset is provided for the needs of this assignment and should not be posted online or submitted to public repositories. The training and test patterns are available on Moodle in ZIP format. More information about this dataset is provided in the Appendix.

You can use any programming language or software library for this assignment. In the labs, we have used MATLAB because it provides well tested functions for building machine learning algorithms that are appropriate for the colonoscopy dataset and the UCI datasets mentioned above.

The assignment is further explained in Sections 2 below. Section 3 of this document gives you an example of how to structure your report and explains the marking scheme. Section 4 presents the deadlines and submission instructions. Section 5 explains the penalties for late submissions, and Section 6 explains how the College deals with plagiarism. Section 7 and Section 8 provide additional information on learning resources and referencing.

2. Implementation and experimentation

You can implement your methods in MATLAB, write your own code, or use a package/library from the internet. You are not tested on programming so the coding style does not have to be perfect and your code does not have to be optimal but it should obviously work correctly. I wouldn't recommend implementing everything from scratch unless you are very experienced with Java, C++, Python or some other programming language or platform. In all cases, make sure that all sources and code taken by others or the internet are cited properly in your Report; otherwise, you may be accused of plagiarism.

Some packages provide techniques for determining the optimal structures of machine learning models automatically as part of the training. In that case, instead of performing experimental tests varying the number of free parameters, as mentioned in Section 1 and in the example provided in the Appendix, these techniques can be used to find the appropriate structure for your model. Still some of these methods may have their own parameters, which require fine tuning.

Note that the performance results of your method would be more meaningful, if a validation technique is used, such as k-fold cross validation (k=7 or k=10 is typically used), or leave-one-out cross validation, or some form of Monte Carlo simulation. Lastly, the use of regularisation, provided in some software packages and in Matlab, normally helps to get better results.

You are expected to test your method using relevant test data and store results in ASCII format for each experiment that you conduct. Results are typically in the form of: number of successfully recognised patterns per class for each experiment; number of unsuccessfully recognised patterns per class for each experiment; the overall average classification success in training and in testing, and average error in training and in testing (e.g. in neural networks the error is typically the squared difference between the actual output vector of the network and the desired one over the whole test set).

The results of your experiments should be stored in ASCII format, in a Jupyter notebook, or in notebook documents produced by other web-based interactive computational environments, specifying whether the result is from the training or the testing phase, and should be submitted together with your report (Moodle will allow to submit additional files, up to 5 files in total can be submitted). Check that these files can be opened and read correctly. Results should be presented using figures and tables and discussed in your Report, i.e. it is not enough just to submit a python notebook or files with results- these are not accepted as a Report submission.

3. Assignment outline and marking scheme

Your work will be presented in a Report (notebook documents are not accepted as a Report). It is important that your Report is properly structured. Sections like the ones shown below should be included in your report to ensure good coverage of the topic. A number of 3000 words, at least, are expected to cover in sufficient depth all aspects of the assignment, but **our marking is not based on the number of words used in the Report**. Also, **you are not just being marked on how good the results of your machine learning algorithm are.** What primarily matters is that you know how to pick the right algorithm and overall approach, explain how they work, and make it work with data. You will need to provide insight on how to (pre)process the data before feeding into the algorithm, how to debug the learned model (not the algorithm), how to measure performance and demonstrate its significance.

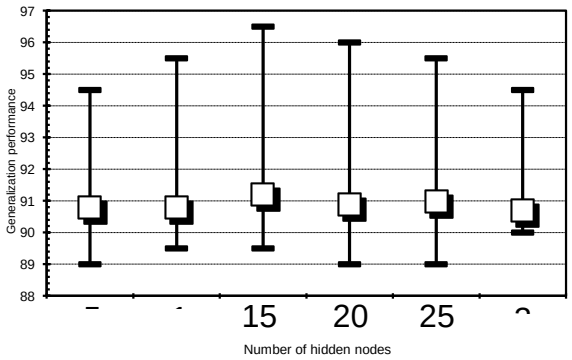
1. Method and methodology used (40% of the mark): the appropriate use and sophistication of the methods and of the overall methodology are marked here

- 1.1 This part should normally describe clearly the method used in your assignment and any relevant parameters (e.g. for neural networks this includes number of hidden nodes, layers, type of activation functions etc), and the rational for using this method. If you are using a particular library or tool, you still need to describe how the method/algorithm that you are using operates. Citing the library, tool, etc., and mentioning the library functions that you have used is not enough to get a high mark.
- 1.2 This part should describe any special techniques/algorithms used as part of your methodology. For example, the algorithm used for training a neural network and its parameters – e.g. if you use Rprop backpropagation then initial learning rate values used should be stated. Also, this part should describe any normalisation techniques used, or other pre-processing or balancing methods, and whether you have used some form of cross-validation, or weight decay, providing details of the particular

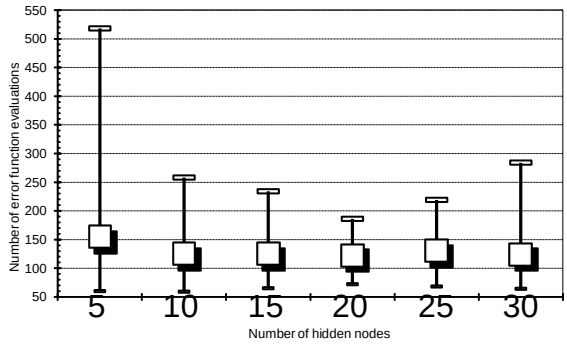
method. Citing the library, tool, etc., and mentioning the library functions that you used is not enough to get a high mark.

2. Experiments, findings and discussion (50% of the mark): you must present and discuss your results. You are expected to run several experiments and calculate basic statistics to summarise performance. Your report must include at least two figures which graphically illustrate quantitative aspects of your results, such as training/testing error curves, performance for sets of learned parameters, algorithm outputs, descriptive statistics, etc.

In this part, you should provide a detailed account of your experiments and results and discuss your findings. You can use Excel or other packages to provide charts - like the figure below, which uses error bars (Box and Whisker Charts in Excel), to show the performance of your algorithm in terms of generalisation. For example, the figure below shows generalisation with respect to number of hidden nodes used in a neural network-based solution. Alternatively, one could use tables to provide the same information by giving for each number of hidden nodes the average value, the minimum value, and the maximum value of generalisation performance (in percentage of successfully recognised patterns) in the tests.



You could also discuss the cost of the computations, e.g. referring to the number of training iterations required or the number of error function evaluations (see figure below for the neural-network based solution discussed above)



In machine learning, overall results are also presented in tables like the one below that shows average performance in terms of recognition success as well as average classification success per class for two methods tested on the same dataset. Confusion matrices can also be used.

Method	Class 1 (%)	Class 2 (%)	Average success (%)
Method 1	83	96	93
Method 2	73	93	88

3. Conclusions (10% of the mark)

3.1 Provide an overview/summary of your work and findings.

3.2 Identify areas for improvement; discuss what you could have done better (particularly important if you failed some of your targets or your results as not as expected).

4. Bibliography: this is not marked but is necessary as it supports the justification of your methodology and methods. Make sure that all sources are cited in the text of the report and

are also listed in the bibliography section- this way you don't get in any trouble with plagiarism detection software.

> Provide a list of the bibliographical/web sources you used. Include publication details and all information necessary to access the online resources. Sources should be cited in the text by (Author name, year) and appear in the references list in alphabetical order by Author's last name. This also applies to websites, e.g. an online article/webpage should be listed in your references; for example:
> (MLOSS, 2011). *Machine learning open source repository. Available online at http://mloss.org/*

NOTE: use of any text or code (even open source code) taken from other sources should be clearly identified and referenced in your report to avoid plagiarism (see Section 6). If you are unsure on which parts of your code needs appropriate referencing do consult the module lecturer.

4. Deadlines and submission instructions

Submission is only through Moodle and consists of the submission of a Report, data files with your results and code (if existing Matlab toolboxes have been used or libraries, these should be mentioned in the Report). Important parts of your code can be included and discussed in the report but a **Notebook document, copy of the Matlab editor etc. is not accepted as a Report submission. The preferred format for the Report is Word document but also PDF or RTF are accepted. You report and the code will be tested for plagiarism.**

Make sure you are familiar with Moodle and able to upload your files (for example you could test the system by uploading a test file). **Hardcopy versions of the report/data/code files will not be accepted.**

You should upload on Moodle the completed assignment by **April 25th, 2019 at 11:00pm** (this is Moodle time not your PC's time. In case you are planning to upload your files whilst at a remote location make sure you check Moodle's time and take into account time zone differences).

Your files should be named according to your last name.

Your Report must have a cover page!
The first page of your report **MUST** have the following information:

Module title and code: Machine Learning- COIY065H7
Name: your first name and last name
Emails: please provide your College email and the email you use- if different from your College email

Your Report should have an Appendix with a description of data files and code submitted; when these are not included in the Report, as appendices, but are submitted separately. Any specific instructions on software use should also be included in an Appendix of the report, entitled "Instructions for using the code".

It is your responsibility to ensure that files transferred from your own machines are in the correct format and that any programs execute as intended on Department's systems prior to the submission date.

Each piece of submitted work MUST also have a page entitled "Academic Declaration" by the author that certifies that the author has read and understood the sections of plagiarism in the document http://www.bbk.ac.uk/mybirkbeck/services/rules/Assessment%20Offences.pdf that describes College's Policy on assessment offences. Confirm that the work is your own, with the work of others fully acknowledged. Submissions must also be accompanied by a declaration

giving us permission to submit your report to the plagiarism testing database that the College is using.

Reports without a Declaration form are not considered as completed assignments and are not marked.

The Academic Declaration should read as follows: "I have read and understood the sections of plagiarism in the College Policy on assessment offences and confirm that the work is my own, with the work of others clearly acknowledged. I give my permission to submit my report to the plagiarism testing database that the College is using and test it using plagiarism detection software, search engines or meta-searching software."

You should note that all original material is retained by the Department for reference by internal and external examiners when moderating and standardising the overall marks after the end of the module. We are aiming to provide grades and feedback through Moodle by June 2nd, 2019.

Those who have any questions or would like to get some early feedback on their assignment before submitting the completed work, they can email <gmagoulas@dcs.bbk.ac.uk> their questions or send a draft of their report for comments. This should be done **by March 22, 2019** as we may not be able to respond to messages in a timely manner after this date.

5. Late coursework

It is our policy to accept and mark late submissions of coursework. You do not need to negotiate new deadlines and there is no need to obtain prior consent of the module lecturer.

The **last day the system will accept a late submission for this module is May 4th, 2019 at 11:00pm** (this is Moodle time not your PC's time. In case you are planning to upload your files whilst at a remote location make sure you check the Blackboard time and take into account time zone differences). **May 4th, 2019 at 11:00pm is the absolute cut-off deadline for coursework submission.**

However, penalty applies on late submissions. Thus the maximum mark one can get in the coursework is 50%. If you believe you have good cause to be excused the penalty for late submission of your coursework, you must make a written request using a mitigating circumstances application form and attach any evidence. Your form should be handed in or emailed to the MSc Programme Administrator (with a carbon copy to the module lecturer and the Programme Director) as soon as possible, ideally that is by the cut-off deadline. This letter/email does not need to be submitted at the same time as you submit the coursework itself but **MUST be submitted by May 14th, 2019.**

Even if the personal circumstances that prevented you from submitting the coursework by the last day (i.e. May 4th, 2019) are extreme, **the Department will not accept coursework after this date**. We will, naturally, be very sympathetic, and the MSc Programme Director will be happy to discuss ways in which you can proceed with your studies, but please do not ask us to accept coursework after this date; we will not be able to as there is a College-wide procedure for managing late submissions and extenuating circumstances in student assessment. As soon as you know that you will not be able to meet the deadline, it will be useful for you to inform the module lecturer. They will be able to advise you on how best to proceed. Another person to speak to, particularly if the problem is serious, is the MSc Programme Director. You will then have the opportunity to discuss various options as to how best to continue your studies.

Further details concerning the rules and regulations with regard to all matters concerning assessment (which naturally includes coursework), you should consult College Regulations at

http://www.bbk.ac.uk/mybirkbeck/services/rules. **Please see the 2018/19 programme booklet for the rules governing Late Submissions and consideration of Mitigating Circumstances and the Policy for Mitigating Circumstances at the College's website** http://www.bbk.ac.uk/mybirkbeck/services/rules.

6. Plagiarism

The College defines plagiarism as "copying a whole or substantial parts of a paper from a source text (e.g. a web site, journal article, book or encyclopedia), without proper acknowledgement; paraphrasing of another's piece of work closely, with minor changes but with the essential meaning, form and/or progression of ideas maintained; piecing together sections of the work of others into a new whole; procuring a paper from a company or essay bank (including Internet sites); submitting another student's work, with or without that student's knowledge; submitting a paper written by someone else (e.g. a peer or relative), and passing it off as one's own; representing a piece of joint or group work as one's own".

The College considers plagiarism a serious offence, and as such it warrants disciplinary action. This is particularly important in assessed pieces of work where the plagiarism goes so far as to dishonestly claim credit for ideas that have been taken from someone else.

Each piece of submitted work MUST have an "Academic Declaration" form signed by the student which certifies that the students have read and understood the sections of plagiarism in the College Regulation and confirm that the work is their own, with the work of others fully acknowledged. Submissions must be also accompanied by a declaration giving us permission to submit coursework to a plagiarism testing database that the College is subscribed.

If you submit work without acknowledgement or reference of other students (or other people), then this is one of the most serious forms of plagiarism. When you wish to include material that is not the result of your own efforts alone, **you should make a reference to their contribution, just as if that were a published piece of work**. You should put a clear acknowledgement (either in the text itself, or as a footnote) identifying the students that you have worked with, and the contribution that they have made to your submission.

7. Referencing

References include the full bibliographic information about the source, such as the author(s)'s name(s), date of publication, title of work, place of publication, and publisher. This information is usually given in the section called Reference List or Bibliography at the end of the text. The key principle is that you should give enough information to allow another person to find the source for themselves.

Here are some examples using the Harvard referencing system:

[when you are referring to a book]
Lewin, K., 1951. Field Theory in Social Science. New York: Harper and Row.

[when you are referring to a chapter in a book, where 'ed.' means editor, and 'edn.' means 'edition']
Piaget, J., 1970. Piaget's theory. In: P. Smith, ed., Handbook of child psychology. 3rd edn. New York: Wiley, 1970, pp. 34-76.

[when you are referring to a journal article]
Holmqvist, M., 2003. A Dynamic Model of Intra- and Interorganizational Learning. Organization Studies, 24(1), 95-123.

[when you are referring to a webpage]

W3C, Web Accessibility Guidelines and Techniques, available online at http://www.w3.org/WAI/guid-tech.html. Last accessed 12/02/2015.

Independent of their type (e.g. book, article, webpage), all references are included at the end of a document in alphabetical order starting from the author's name as in the example above.

8. Useful resources

Here are some resources on plagiarism, study skills, time management and referencing that can help you to better manage your project and avoid plagiarism.

On Plagiarism

- https://owl.purdue.edu/owl/teacher_and_tutor_resources/preventing_plagiarism/index.html

On Referencing Systems

- Harvard guide to citing references Available to online at: http://www.open.ac.uk/libraryservices/documents/Harvard_citation_hlp.pdf

On Study Skills

- http://www.bbk.ac.uk/student-services/learning-development

APPENDIX

Detection of lesions in colonoscopy: the problem and the data

Colonoscopy is the most accurate screening technique for detecting polyps, also allowing biopsy of lesions and resection of most of the polyps. The procedure is carried out by an expert who interprets the physical surface properties of the tissue- such as the roughness or the smoothness, the regularity, and the shape - to detect abnormalities. Adjacent surfaces of the colon lining showing different properties are distinguished on the basis of the textural variations of their tissue (the texture of the tissue is considered as a composition of pit patterns). These textural alterations of the colonic mucosal surface signify that this property could also be used for the automatic detection of lesions.

The approach followed to generate the dataset consists of two processing stages. The first stage consists of procedures that are applied to the frames of the video sequence to extract all the identifiable features, which form the feature vectors. To this end, a family of texture attributes that correspond to the components of the feature vectors and account for the main spatial relations between the grey levels of the texture has been chosen. The particular method applied is called concurrent matrices and produces 16-dimensional features vectors for each region (rectangular windows of 64 x 64 pixels) of the frame is applied on.

The second processing stage, which is the focus of this assignment, decides on the image regions characterisation. To this end, several researchers have proposed techniques that range from linear discriminant analysis to sophisticated AI detection schemes that are based on artificial neural networks, support vector machines, multiple-classifier, fuzzy or neuro-fuzzy systems.

For this assignment, you are provided with a set of data files that contain patterns (feature vectors) extracted from a short endoscopy video sequence. The dataset is organised into training and test data files; there is a training data file and a test data file for each frame of the video sequence.

Files with training data are named as `framenumber`trn.ssv (e.g. 1trn.ssv, 4trn.ssv), whilst files with test data are named as `framenumber`tst.ssv (e.g. 1tstn.ssv, 4tst.ssv). These are ASCII files that contains train/test data and can be opened using a standard editor, e.g. Wordpad, MS-Word, or using MS-Excel.

In these files, each row represents a training or testing pattern and consists of 17 elements (columns). The first 16 elements correspond to features of a particular frame region without any normalisation. The last element is either 0 or 255: a value of zero indicates that the corresponding vector represents a normal tissue sample, while a value of 255 indicates that the pattern is associated with an abnormal tissue sample, and if you want to use it in your algorithm, typically, you should substitute it by 1.

This should normally lead you to use 16-dimensional vectors for input data and 2-dimensional vectors for the desired output (formulating the problem as a 2-class problem, i.e. normal/abnormal defined as 0/1). Typically a normalisation procedure should be applied on the training and testing data to bring their values in the range of [0,1] or [-1,1] depending on the application.

For training, you should use vectors from the `framenumber`trn.ssv files and for testing vectors from the `framenumber`tst.ssv.

Further details on the problem and the use of classifiers in endoscopy can be found in the paper:

Magoulas G.D., Plagianakos V.P., and Vrahatis M.N., Neural Network-based Colonoscopic Diagnosis Using On-line Learning and Differential Evolution, Applied Soft Computing, Vol. 4(4), 369-379, 2004. Available online at: http://www.dcs.bbk.ac.uk/~gmagoulas/ApplSoftComp.pdf

Experimental investigation on the performance of your classifier

Experimentation can take different forms and next we provide some examples. A classifier is normally trained to achieve up to 97% of success in discriminating between normal/abnormal patterns (use data from the frames stored in the "?trn.ssv" files). For example, if the training set has 1200 (say 600 normal + 600 abnormal) patterns then we consider that a neural network has been trained successfully when it is able to categorise at least 1164 out of the 1200 patterns. When real-world noisy data are used a lower success of 90% or even lower, e.g. 80%, in training might also be considered as a good result in testing – this depends on the level of noise in the data and it is difficult to know in advance, i.e. before doing some attempts to train the classifiers. The trained classifier is tested over the entire test set (use data from the frames stored in the "?tst.ssv" files).

Although you can use any machine learning classifier you wish, below we provide an example of how neural networks could be applied to this problem. A neural network can be trained by minimising a measure related to network's performance, such as the learning error which can be defined by the sum of the squared difference between the actual output vector of the network and the desired one over the whole training set. This approach is very popular in neural network training and includes learning algorithms that operate off-line, also called batch learning, or on-line learning, also called pattern-based learning.

For example, if you use feedforward or recurrent networks as classifiers, you could try the following:

1) Vary the number of hidden nodes from 5 to 55 in steps of 10 (i.e. 6 hidden node configurations in total). For each number of hidden nodes, use the frames training files to train 10 networks starting from the same random initial weights to get an estimation of the performance. If you use Matlab, you can use any of the following learning algorithms: Scaled conjugate gradients (SCG), Levenberg- Marquardt and Rprop (all these learning algorithms are already available in Matlab). Store in ASCII format for each frame: the number of successfully recognised normal patterns, the number of unsuccessfully recognised normal patterns, the number of successfully recognised abnormal patterns, the number of unsuccessfully recognised abnormal patterns, the overall classification success in training, the number of iterations required (epochs) to reach the training goal (ideally 90% success in training) and the sum of the squared error in training (i.e. the sum of the squared difference between the actual output vector of the network and the desired one over the whole training set).
2) Test each one of the above trained classifiers with the test data of each frame and store in ASCII format for each frame: the number of successfully recognised normal patterns, the number of unsuccessfully recognised normal patterns, the number of successfully recognised abnormal patterns, the number of unsuccessfully recognised abnormal patterns, the overall classification success in testing, and the sum of the squared error in testing (i.e. the sum of the squared difference between the actual output vector of the network and the desired one over the whole test set).

The dataset on Moodle contains several frames so you can do as described above for each frame, or you can combine all frames' training data into one file and use it for training, and then use the frames' test data for testing. In any case, try to create balanced datasets for training (use almost equal number of normal and abnormal patterns); because training using imbalanced data is more difficult and typically produces poor results in testing.