Imperial College London

Department of Mathematics

MATH96007 - MATH97019 - MATH97097
Methods for Data Science
Years 3/4/5

Coursework 2 – Random Forests, SVMs, Neural Networks

Deadline: Monday, 2 December 2019, 4 pm

General instructions

The goal of this project is to analyse a data set using some of the tools introduced in the lectures, but also following your own initiative. Coursework tasks are different from exams: they can be more open-ended and may require going beyond what we have covered explicitly in lectures. Initiative and creativity are important, as is the ability to pull together the course content, draw new links between subjects and back up your analysis with relevant computations. The quality of presentation and communication is very important, so use good combinations of tables and figures to present your results.

Submission instructions can be found at the end of this document. You must submit a Jupyter notebook, execute it, and submit the html file. You can produce a notebook in Google Colab, or you are free to use your local Jupyter notebook through the Anaconda environment downloaded on your computer. Before you produce your html, make sure all cells are executed, so their output appears in your html file.

To gain the marks for each part you are required to: (1) complete the task as described; (2) comment any code so that we can understand each step; (3) provide a brief written introduction to the task explaining what you did and why you did it; (4) provide appropriate, relevant, clearly labelled figures documenting and summarising your findings; (5) provide an explanation to your findings in mathematical terms based on your own computations and analysis and linking the outcomes to concepts presented in class or in the literature. Marks will be reserved and allocated for: presentation, quality of code, clarity of arguments, and additional *relevant* work that shows initiative and *understanding* beyond the minimal task stated in the coursework.

This coursework is worth **35% of your total mark**, and contains a **mastery component** for MSc and MSci students.

Overview

In this second coursework, you will work with a data set that contains various variables describing used cars. After your studious and successful time at Imperial College, you landed your dream job as a used car dealer. Actually, you do not know much about used cars, but, luckily, you have a record of the decisions made by your predecessor where she had rated a series of cars as `unacceptable', `acceptable', `good', or `very good'. Fortunately, your predecessor also kept detailed records to support these decisions based on the following **descriptors**:

- *buying:* buying price
- *maint*: price of the maintenance
- *doors:* number of doors
- *persons*: capacity in terms of persons to carry
- *lug_boot:* the size of luggage boot
- *safety:* estimated safety of the car

In an effort to impress your manager, you will compare different classification methods that can work with such multi-dimensional data.

You can find the data on Blackboard.

Preliminaries

You should do some data exploration, perhaps reusing some of the code from CW1. This can help inform your explanations below but **you do not need to report your data exploration in this CW.**

1. You are expected to **prepare and clean up the data** sufficiently without our specific instructions. Just **write down the Python commands** you used for the preparation of the dataset.
2. A key step in your pre-processing is the **standardisation of the descriptors of the data set,** which you can do using sklearn. Explain briefly why this is needed and write down your commands.

On Blackboard you are provided with a training set (containing 80% of the data) and a test set (containing 20% of the data). Throughout the coursework below:

- You should optimise hyperparameters using **stratified k-fold cross validation** of your training set. Make sure that the k-folds are properly balanced by using the correct sklearn options.
- Do **not** use your test set for the optimisation of hyperparameters. The test set should only be used to test the final generalisability of your optimised model.

Task 1: Random Forest Classifier (25 marks)

Train a random forest classifier on the training set to optimise the **accuracy** of your validation predictions using 5-fold stratified cross validation.

You should use the 5-fold cross validation to explore and optimise over suitable ranges the following hyperparameters: (i) number of decision trees; (ii) depth of trees, (iii) maximum number of descriptors (features) randomly chosen at each split.

Note: Although other measures of performance, such as precision, recall, etc, could (and should) be used for comparison, here we only concentrate on accuracy for simplicity.

Explain and document your choice for the best RF model. Explain which of the hyperparameters have a bigger impact on performance. Base your answer on computations and explain in relation to the mathematical basis of the algorithm.

Task 2: Support Vector Machines (25 marks)

Train a support vector machine (SVM) on your training data using three different kernel functions: (i) linear; (ii) polynomial, (iii) RBF. Each of those kernels have a few hyperparameters which you should explore and tune using 5-fold cross validation to find the kernel (and hyperparameters) with the highest validation accuracy.

Explain and document your choice for the best SVM model (kernel and hyperparameters) for our data. Base your answer on computations and give explanations related to the mathematical basis of the algorithm.

Task 3: Neural Networks (30 marks)

In Task 3, you will explore how a feed-forward neural network performs on the training set.

Using Pytorch, train a neural network to classify each car in the training set based on the given descriptors.

Setup of the network: Your network should have two hidden layers, each with 200 neurons. You should use ReLU as your activation function. Fix the optimisation method to be stochastic gradient descent (SGD), and define the loss function as cross-entropy. You should train on batches of 64 data points with a learning rate of 0.01 for 120 epochs.

3.1 Show and document how changing the **learning rate** to: (i) 0.0005 and (ii) 0.95 leads to poor convergence.

3.2 Show and document how changing the **batch size** to: (i) 2 and (ii) 256 leads to poor convergence and performance. Explain the reasons.

3.2 In Pytorch, implement **dropout regularisation** to the second layer of the NN, and tune the dropout rate to optmise the validation accuracy of the NN.

Explain and document your choice for the best NN model (dropout rate) for the given architecture basing your answer on computations and on explanations related to the mathematical basis of the algorithm.

Task 4: Discussion (20 marks)

4.1 Compare the performance of the three classifiers you have obtained in Tasks 1, 2 and 3 by applying them to the **test data set.** You should report the accuracy, recall, precision, and F1 score and any other relevant score derived from the **confusion matrix**. Examine your results in relation to the performance obtained for the training set.

4.2 Discuss the suitability of each of the three methods for our task. Comment on their generalizability to the test data, computational cost, and the appropriateness given the dimensionality of the data and any other insights based on your study of the descriptors of the data set. You should base your evaluations on evidence and computations.

Task 5: *(for MSc and MSci students only)* (25 marks)

5.1 **(20 marks)** Consider the NN model you implemented in Task 3. Optimise a new NN where the architecture is changed to have **5 hidden layers with 80 neurons each**. Compare the performance of this 'deep' network to the 'shallow' network in Task 3 in terms of performance, training and computational cost. Feel free to provide additional evidence and computations to support your analysis.

5.2 **(5 marks)** Consider the NN model you implemented in Task 3. Optimise a NN with the same architecture as in Task 3 but changing the activation units from ReLU to **Sigmoidal**. Compare the performance of the ReLU and sigmoidal NNs and, specifically, their speed of convergence.

Submission instructions

You will upload two documents to Blackboard, wrapped into a single zip file:

1) Your notebook as an ipynb file.
2) Export your notebook as an html file.

You are also required to comply with these specific requirements:

- Name your files as ‘SurnameCID.zip’, e.g. Smith1234567.zip. Do not submit multiple files.
- Your ipynb file must produce all plots that appear in your html file, i.e., make sure you have run all cells in the notebook before exporting the html.
- Use clear headings in your html to indicate the answers to each question, e.g. ‘Task 1’.

To avoid last minute problems with your online submission, we recommend that you upload versions of your coursework early, before the deadline. You will be able to update your CW until the deadline but this provides you with some safety back up.

Note: *There seem to be some issues with particular browsers (or settings like cookie or popup blockers) when submitting to Turnitin. If the submission 'hangs', please try another browser. You should also check that your files are not empty or corrupted after submission.*

Needless to say, projects must be your own work.

You may discuss the analysis with your colleagues but the code, writing, figures and analysis must be your own. The Department may use code profiling and tools such as Turnitin to check for plagiarism, and plagiarism cannot be tolerated.

Copying and plagiarism, if they occur, may force the Department to stop offering project-based courses such as this one.