AERO40041 Coursework 2 – Machine learning

alex.skillen@manchester.ac.uk

November 2024

1 Introduction

In this coursework, you will implement 1) Stochastic Gradient Descent, 2) a mean absolute error loss function trained with gradient descent, and 3) a small neural network. The coursework is released in week 7, but some topics will not be covered until later. It is suggested you attempt each task *after* the corresponding material has been covered in the lectures.

The labs are designed to provide supporting materials. Participation in these will significantly help you in the coursework. In the labs, code is given that you can use and adapt. You may adapt that code directly (i.e. it will not be considered plagiarism to use the code I've provided and modify it according to the question).

1.1 Permitted libraries

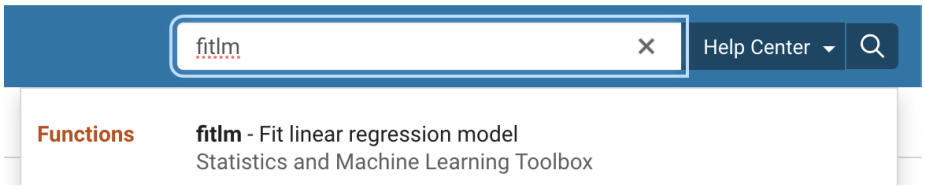
You may attempt this coursework in Python (recommended), or Matlab. Naturally, there will be no difference to the mark scheme no matter which language you choose. In either case, the aim is to implement the algorithms 'from scratch' rather than to use pre-existing libraries that have implemented the algorithms for you. To this end, in Python, you may import `numpy` (for maths operators), `Pandas` (for reading CSV files) and `matplotlib` (for plotting charts) only.

In Matlab, you may not use any functions that are part of external toolboxes (especially not the 'Statistics and Machine Learning Toolbox', the 'Curve fitting toolbox' or the 'Deep learning toolbox'). To test if a function is in a toolbox, you can go to the help page here:

`https://uk.mathworks.com/help/matlab/`

Type the function name in the search box (at the top right of the page) but do not press enter. If the function is part of a toolbox, you will see the toolbox listed below the function name and description, as shown in Figure 1 (a). Conversely, if the function is not part of a toolbox, then you will see the word 'MATLAB', indicating the function is part of the base Matlab implementation, as shown in Figure 1 (b). If it feels like you have not implemented the method from scratch (using linear algebra and standard mathematical operators only) then you're probably not. Please ask for clarification on what is permitted and what is not if anything is unclear.

a)



b)

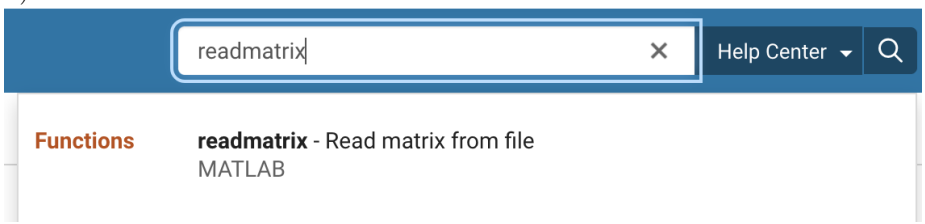


Figure 1: Using the Mathworks website to check if a function belongs to a toolbox. Top (a): A function belonging to the Statistics and Machine Learning Toolbox – not okay to use. Bottom (b): A function belonging to base Matlab – fine to use.

1.2 Groupwork

You may work in pairs, or on your own. It is up to you. However, you may not work in groups larger than two persons (you plus one other). The pair need not be the same as the one you had defined in Coursework 1.

If you work in pairs, you need to add a small declaration at the start of each file submitted as a code comment, for example:

#This submission was prepared by [Name1] [ID1] and #[Name2] [ID2].

If you do work in pairs, you must contribute equally and must each understand all parts of the code submitted. You also need to register your group on Blackboard before Monday the 18th of November. Click on groups (under the discussion board link), then 'View sign-up Sheet', pick a group number, sign up, and get your partner to do the same. Please do not join a group with another person already in it unless you both wish to work as a pair and you have discussed this with them. If you face any issues setting up your group, please let me know and I will fix it. Please register your group even if its membership is the same as for CW1.

If you are working individually, you do not need to sign up.

2 Linear regression

For the tasks in this section, you may use the code provided in '`LinearRegression.ipynb`' to guide you. You may adapt that code directly (i.e. it will not be considered plagiarism to use the code provided and modify it according to the question). You may also port it to Matlab if you prefer to do the coursework in Matlab over Python.

Your tasks are as follows:

1. Implement stochastic gradient descent for a linear regression model that works on the window heat loss dataset. The filename you use for submission should be LinearRegression_SGD{.py | .m | .pdf}. Marks will be awarded as follows: Code runs and produces the correct output (6 marks). The `epoch` counter has the correct meaning (4 marks). Total 10 marks.

2. Change the provided example in '`LinearRegression.ipynb`' from Mean Square Error (MSE) loss to Mean Absolute Error and train it on the window heat loss dataset with gradient descent. The filename you use for submission should be LinearRegression_MAE{.py | .m | .pdf}. Marks will be awarded as follows: Loss function correctly implemented (4 marks). (Stochastic) Gradient descent is correctly implemented. You may use Gradient descent or Stochastic Gradient descent – either is fine. (6 marks). Total 10 marks.

3 Neural Network for classification

For the tasks in this section, you may use the code provided in '`NeuralNetwork.ipynb`' to guide you. You may adapt that code directly (i.e. it will not be considered plagiarism to use the code provided and modify it according to the question). You may also port it to Matlab if you prefer to do the coursework in Matlab over Python.

Your task is to implement a classification neural network. Your network should take two features as input, x_1 and x_2, and predict the class for other unknown x_1 and x_2. The dataset you will train your network on is plotted below (filename 'classification_training.csv'). You also have test data in 'classification_test.csv'

You will train a network with tanh activation functions for hidden layer(s), and sigmoid activation for the output. (40 marks for a working implementation without errors)

You should perform a small hyperparameter search to determine appropriate values for the learning rate and number of neurons in the hidden layer. (10 marks for appropriate choices for hyperparameters.)

Your code should output the decision boundary for your network (your final model with appropriate hyperparameters). Hint: to compute the decision boundary, you might discretise the parameter space into a grid and perform a forward network evaluation of the trained model at each point on the grid,

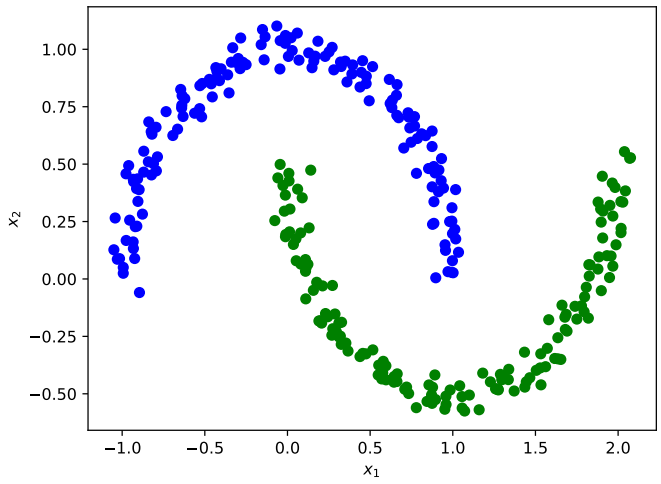


Figure 2: Training dataset for Neural Network classification problem. Features are x_1 and x_2. The label is denoted by the point colours (blue is $y = 0$, green is $y = 1$).

plotting the result as a contour map. You will want the resolution of the grid to be sufficiently fine so that smooth contours are produced and refining it further does not significantly alter the plot produced. The function `'meshgrid'` is useful for this (available in both Matlab and Python) (10 marks).

Your submission for this task should have the filename NeuralNetwork_classification{.py | .m | .pdf}.

4 Discretionary marks

An additional 20 marks are available across all tasks for high-quality code that is professionally written. Examples of the things that might attract marks in this category are:

- Does the code have an efficient implementation to solve the problem?
- Is the code well-formatted and clear?
- Are variable and function names meaningful?
- Are comments used appropriately to explain complex logic or non-obvious code sections? Note you should not provide excessive comments, especially for simpler parts of the code.

- Have you ensured there are no unnecessary computations or redundant operations?
- Does the code exhibit creativity in the solution, such as finding an elegant or efficient approach?
- Can your neural network code generalise to different network architectures?

This is a non-exhaustive list, and marks may be awarded for other things at our academic judgement.

5 What to submit?

You are not required to write a report or provide any other information. You only need to submit the codes requested. For each task, you will submit the code in two forms: 1) a Matlab .m file or a Python .py file as described above. 2) a PDF file containing a printout of your code *and* the output of the code such as any plots or printouts the code produces when it is run (for example, the plot of the decision boundary of the neural network).

You must submit the same code in the PDF as is in the .m file or .py file. The reason we ask for the code in two forms is to help with marking (a PDF is easier to open in Blackboard, but sometimes we might need to download the code and run it if we're unsure if/how it works).

Further instructions on submitting your files (either as a pair or individually) will be provided soon (before week 11).

6 Similarity reports

Similarity checks will be performed against any code in the checking tool's database. You cannot take code from the internet. You must write your own code or adapt the provided code. Note that you may receive a high similarity score using the sample code I gave. This is nothing to worry about.

7 Due date

This coursework should be submitted by Wednesday of Week 12 (i.e. the 11th of December 2024), 6 pm (UK time).