Imperial College
London

MLNC – Machine Learning & Neural Computation – Dr Aldo Faisal

Assessed Coursework 1

To be returned via TurnItIn as indicated online.

The focus of this first of two assessed courseworks is supervised learning using regression. This coursework counts for $\frac{1}{3}$ of your final course mark.
The coursework submission should comprise a report and two Matlab files. Your coursework report should contain on each page in the header: Your name, your ic.ac.uk email, your CID and your degree course (i.e. "UG", "MSc NT", "MSc HBR", "MRes NT") on the header of each page. Your text should provide *brief* explanations so that the GTA markers can understand what you did. Please use *succinct* answers to the questions. The final document should be submitted in **PDF** format (we recommend using Latex but Word or other word processors are also fine). Figures should be from Matlab or a drawing program and appropriately labelled, captioned and referenced in the main text. Poor, unclear or incomprehensible presentation may result in a loss of marks. Marks are shown next to each question. Note that the marks are only indicative.
Your coursework report should not be longer than 3 single sided A4 pages including all figures (excluding code appendices). You are encouraged to discuss with other students, but your answers should be *yours*, i.e., written by you, in your own words, showing your own understanding. You have to produce your own code and submit it where appropriate.

You will need to write Matlab code, there please paste the full annotated source code that generated your results in the appendix of your PDF submission, in addition please provide the two files `trainRegressor.m`, `testRegressor.m`. You are allowed to use all built-in Matlab functions and any Matlab functions supplied by the course or written by you. If your Matlab functions do not pass the automatic tests performed we will be unable to automatically evaluate their performance and may deduct marks.
Download and open the coursework zip-file from Blackboard. All coursework has to be submitted individually and will be marked as such. You are welcome to discuss your work during the lab sessions with other students and your GTAs. All code files must contain a comment line identifying the Name and CID of the submitter.
How to submit: Your 3 submission files (`trainRegressor.m`, `testRegressor.m`, and your report PDF) should be placed into a new single folder. The folder name should be your CID without leading "0"s. Go into the parent directory and zip your submission folder into a file called "YourCID.zip" . Double-check that when you unzip your "YourCID.zip" file a folder called "YourCID" is created and the files are within. The submission zip file should then be uploaded to the TurnItIn submission box on Blackboard. Adhering to the correct format is important so the GTAs can run auto-testing of your models and see how they compare against each other.

Supervised learning of 1-bedroom/studio rents from London location

You have been provided with a dataset that contains thousands of rental prices (pcm) for single bedroom properties in London along with corresponding Lat/Long coordinates. This is a real dataset collected using automated scripts from online sources, thus there may be a number of properties that are not located within Greater London and data clean up is needed. Also provided is a set of coordinates for all stations on the London Underground network. The aim of this coursework is to use supervised learning techniques from the course to learn from the

supplied sample data the mapping from geographical location, specified as two-dimensional vector x = [latitude longitude], to rental costs per calendar month y = [GBP], i.e. identify the unkown $f(x) = y$. Thus, for an arbitrary location in Greater London we want to know the predicted rental price from your machine learning system. You can choose any of the methods discussed in the course, provided that you implement them yourself from first principles without using specialist toolboxes or external code.

In your coursework zip directory you will find the following files:

- `.mat` file contains the location and price data that you will be using for the exercise, stored as a structure named `Prices`, with two fields `rent` and `location` (in latitude and longitude). In addition it also contains a structure named `Tube`, which contains two fields `station` and `location` with all London Underground stations.
- `trainRegressor.m` – an empty stub file you have to write in
- `testRegressor.m` – an empty stub file you have to write in
- `sanityCheck.m` – a function that will check whether your code follows specifications
- `heatmapRent.m` – a function that produces uses your model to produce a heatmap of rental prices
- `personalisedTube.m` – a function that converts your CID into a *faux* home tube

Your overall goal is to write Matlab functions that perform machine learning, so as to be able to answer the subsequently described subquestions and reason on your design choices, how you setup, train, test and validate your model. In the following sub-questions you will be asked to generate plots from data or your results. Plots that do not have appropriate labels (i.e. the name of dimension and the units in square brackets, e.g. "Rent [GBP]") will not be counted. Therefore, figures should be exported so that lines and data points are clearly visible when exported and any text in the figures is clearly readable. Use "export figure" to export the figure from Matlab and store it as PNG. Untidy or "grainy" figures will not be considered.

1. Visualise in a 3D plot (`plot3`) the rental price raw data for Greater London (rotate the view appropriately to be as informative as possible). (5 points)
2. Write the function (`trainRegressor.m`) and train your system with the data set. This sub-question carries the bulk of your design choices on how to model and predict your data – you may experiment with approaches and parameters, but in the end submit only your "definitive" version of how you want perform the training that you have to justify as being the most suitable for this problem given the methods you have leaerned. Explain and discuss your solution approach and how you chose any parameters that you set as constants. (35 points)

   - `trainRegressor.m` – a Matlab function that accepts training inputs `trainIn` (a two-column matrix of Longitude and Latitude pairs, where the rows correspond to different training data points) and training outputs `trainOut` (a column vector of rent prices, where each row rental price corresponds to the rental location in `trainIn`). The function is to return a single variable `params` (potentially a structure or matrix depending on your design choices).
     `>>params = trainRegressor ( trainIn, trainOut )`
     Your code should be documented with sufficient comments so another reader is able to follow why and what you did. Any free parameters that you do not want or cannot learn from the provided date set, have to be set as constants within the trainRegressor function. All your training code should sit inside this function, no additional files should be submitted to enable model training.

3. Test your predictions using the Root Mean Square Error (RMSE) of your rental price predictions as metric. This sub-question carries the bulk of your design choices on how to test and validate your work – you may experiment with approaches and parameters, but in the end submit only your "definitive" version

of how you want perform the testing and validation that you have to justify as being the most suitable for this problem given the methods you have learned in the course.

You will have to choose how you split the data, how many folds to use etc. Report your findings and discuss them briefly. To this end write `testRegressor.m`. Hint: acceptable solution must have a RMS lower than the average rental price across all our rental data. (35 points)

- `testRegressor.m` – a Matlab function that accepts testing inputs `testIn` (a two-column matrix of Longitude and Latitude pairs, where the rows correspond to different test data points) and `params` (the parameter variable created by trainRegressor). The function should return a column vector of rental prices for the corresponding locations called results.
  `>>results = testRegressor ( testIn , params )`
  All your testing code should sit inside this function, no additional files should be submitted to enable model testing.
- Use `sanityCheck.m` to check your two functions (`trainRegressor.m`, `testRegressor.m`) match the specifications we use. Use the function with the following command in Matlab:
  `>>sanityCheck ( @trainRegressor ,@testRegressor )`
  Note: If your functions do not pass the automatic tests performed by this function we will be unable to automatically evaluate their performance and may deduct marks.

4. Visualise the heatmap of rental prices. To this end call `heatmapRent(@testRegressor, params)` and show your heatmap.

   - `heatmapRent(@testRegressor, params)` – a Matlab function that takes your `testRegressor` function that you have written, called as function argument "@testRegeressor" (hence prefixed with an "@" symbol) and a `params` variable, which has to have the same format as the `params` variable used by `testRegressor`.

   Discuss briefly to what extent the heatmap you obtain captures elements of the data set and to what extent it captures artefacts of your modelling choices. (5 points)

5. What is the predicted price if you could rent at your *faux* home tube stop. Your *faux* home tube stop can be determined using the function `personalisedTube(CID)`, where `CID` is your college ID number without leading zeros. Report the results and briefly discuss your findings. (20 points)

Hints:

1. Start with a simple regressor and testing/validation approach and get it to work (step 2 & 3), so that you can get quickly e.g. to visualise the heatmap and rental price prediction (step 4 & 5). This will help you score sufficient points to pass the coursework and, in addition, it may inform your model design and testing choices. You can then improve your design and testing approaches. This is a standard computer engineering approach called "successive refinement", that will quickly suggest if adding improvements provide a boost or only marginal benefits. I.e. in Machine Learning "speak" it prevents overfitting of the coursework questions by the student by regularising the time invested.

2. If you have questions about the coursework please make use of Labs or Piazza. You will understand that GTAs cannot provide you with answers that directly solve the coursework, however they will be delighted to help you with conceptual, programming, mathematics questions underlying your coursework.