**Assessment 2: Machine Learning and Optimisation**

This assignment contributes 70% of the overall module mark for COMP2002 and is an individual assignment. You must submit the deliverables to the DLE by the specified submission dates. The coursework has two parts – one is a machine learning exercise and the second is about evolutionary computation. You must complete and submit both parts. Each part is worth 50% of the coursework mark. Both a Jupyter notebook and RMarkdown file have been placed on the DLE for you to use. You should download one of them and use it to implement the code you need to complete the tasks below.

**PART 1 – MACHINE LEARNING**

You have been provided with datasets relating to Health Insurance Premiums. Your task is to train regression models that predict the type of cover and the cost of the premiums based on 9 inputs. You must complete the following tasks:

**Task 1.1 – Data preparation (10% of total mark)**

The first phase of the work requires you to load the data you have been provided with into your selected program. Before the data can be used to train and test your models you must first prepare it – this means that the inputs must be converted to a suitable format. There is no missing data in the dataset.

**Task 1.2 – Regression (20% of total mark)**

Having prepared the data you must now build a regression tool that can predict new points. Use the following regression implementations within the respective program packages to construct predictors for the dataset:

• Random Forest

• Neural Network

• Support Vector Machine

You must demonstrate that each regressor is capable of providing a prediction for a given input.

**Task 1.3 – Assessment of regression (20% of total mark)**

The regression models you have used in the previous task must be assessed. To do this you are required to assess the mean square error rate for each model. You may use the MSE implementation available to do this. It is not sufficient to report a single MSE rate. You must use cross-validation to report training and testing results and report these values using a boxplot. You will also need to write a summary analysing your results and findings.

**PART 2 – OPTIMISATION**

The second part of this assignment requires you to implement an optimiser to solve a timetabling problem for a hospital. You have been provided with a file that describes surgeons and lists the number of surgeries that can be performed in a day and whether an anaesthetist is required. Your task is to optimise a timetable for one day of surgery in such a way that timetable constraints are minimised. There are 9 available time slots per day. You have three operating theatres and two anaesthetists available.

Relevant constraints for this assessment are:

* A surgery cannot be scheduled for a time when the surgeon is in another surgery (concurrence constraints).
* A surgeon cannot perform two surgeries in a row (precedence constraints).

You have been provided with a file describing 5 surgeons, the number and type of surgery, and if an anesthetist is required. You must design and implement a fitness function by taking the number of the concurrence constraints and multiplying them with the number of precedence constraints. This fitness function should be minimised – the ideal timetable is one with no constraint violations at all, in which case the function will return 0.

**Task 2.1 – Generation of random solutions (10% of total mark)**

Your first task is to implement the fitness function. Your code should read the file provided and given a timetable it should return the quality of the schedule in terms of the constraint violations, as described above. You should call your function and print out the fitness of a random route.

**Task 2.2 – Algorithm implementation (25% of total mark)**

You should implement a hillclimber (as described in the lectures) to optimise the problem implemented in Task 2.1. Your algorithm must have the following features:

* It should be possible to use one of two mutation operators – one (surgery replace) mutates a solution by placing a surgery into another slot in the timetable, and the other is a ruin-and-recreate operator (which generates a completely new solution at random).
* At each iteration your single parent solution should be used to generate a single child operator by using one of the mutation operators (either the swap or the ruin-and-recreate for the entire run of the algorithm).
* At the end of an iteration the algorithm should retain the parent or child that has the best fitness.
* At the end of an iteration the best (parent or child) solution’s fitness should be added to a list of the best fitnesses, which is returned along with the best solution at the end of the optimisation.

**Task 2.3 – Visualisation of results (15% of total mark)**

You should run the algorithm twice for 500 iterations – once for each mutation operator. Repeat this 30 times, so that you get 30 fitness lists for the swap operator and 30 fitness lists for the ruin-and recreate operator. Plot the average, maximum, and minimum fitness at each iteration for each operator. You should plot them on the same graph so that they can be compared. You should be able to see which optimiser is best. You need to summarise your results, stating which is the preferred method and why.

**COURSEWORK DELIVERABLES**

Both a Jupyter notebook and RMarkdown file have been provided on the DLE for you to use for this coursework. You should implement your code in it and submit it to the DLE ahead of the deadline specified in the submission dates earlier in this document. Please indicate which task each section of the notebook refers to using a Markdown cell.

Please check your submitted files are correct by downloading them again and checking that they work. You will receive a confirmation receipt by email when your work has been properly submitted – if you do not receive this email then your work has not been submitted.

**ASSESSMENT CRITERIA**

Your work will be assessed according to the rubric found in Tables 2 and 3. Your mark for this piece of coursework will be based on an aggregation of the marks for each category. Marks will be awarded based on both the demonstration and the report.

A close-up of a list of text

Description automatically generatedA screenshot of a computer

Description automatically generated