COMP3411/9814 Artificial Intelligence
Summer session, 2020

Assignment 2 – Heuristics, Search, Learning

Due: Wednesday 05 February, 11:59pm

Marks: 20% of final assessment for COMP3411/9814 Artificial Intelligence

Question 1: Search Algorithms for the 15-Puzzle

In this question you will construct a table showing the number of states expanded when the 15-puzzle is solved, from various starting positions, using four different searches:

(i) Uniform Cost Search (with Dijkstra's Algorithm)

(ii) Iterative Deepening Search

(iii) A* Search (using the Manhattan Distance heuristic)

(iv) Iterative Deepening A* Search

Go to the Course Web Site, Week 3 Prolog Code: Path Search, scroll to the Activity at the bottom of the page and click on "`prolog search.zip`". Unzip the file and change directory to prolog search, e.g.

```
unzip prolog_search.zip
cd prolog_search
```

Start prolog and load puzzle15.pl and ucsdijkstra.pl by typing

```
[puzzle15].
[ucsdijkstra].
```

Then invoke the search for the specified start10 position by typing

```
start10(Pos),solve(Pos,Sol,G,N),showsol(Sol).
```

When the answer comes back, just hit Enter/Return. This version of UCS uses Dijkstra's algorithm which is memory efficient, but is designed to return only one answer. Note that the length of the path is returned as G, and the total number of states expanded during the search is returned as N.

a) Draw up a table with four rows and five columns. Label the rows as UCS, IDS, A* and IDA*, and the columns as `start10, start12, start20, start30` and `start40`. Run each of the following algorithms on each of the 5 start states:

(i)[ucsdijkstra]
(ii)[ideepsearch]
(iii)[astar]
(iv)[idastar]

In each case, record in your table the number of nodes generated during the search. If the algorithm runs out of memory, just write "Mem" in your table. If the code runs for five minutes without producing out- put, terminate the process by typing Control-C and then "a", and write "Time" in your table. Note that you will need to re-start prolog each time you switch to a different search.

b) Briefly discuss the efficiency of these four algorithms (including both time and memory usage).

Question 2: Heuristic Path Search for 15-Puzzle

In this question you will be exploring an Iterative Deepening version of the Heuristic Path Search algorithm discussed in the Week 2 Tutorial. Draw up a table in the following format:

	start50		start60		start64	
IDA*	50	14642512	60	321252368	64	1209086782
1.2 1.4 1.6						
Greedy						

The top row of the table has been filled in for you (to save you from running some rather long computations).

(a) Run [greedy] forstart50, start60 and start64, and record the values returned for G and N in the last row of your table (using the Manhattan Distance heuristic defined in puzzle15.pl).

(b) Now copy idastar.pl to a new file heuristic.pl and modify the code of this new file so that it uses an Iterative Deepening version of the Heuristic Path Search algorithm discussed in the Weak 2 Tutorial Exercise, with w = 1.2 .
In your submitted document, briefly show the section of code that was changed, and the replacement code.

(c) Run [heuristic] on start50, start60 and start64 and record the values of G and N in your table. Now modify your code so that the value of w is 1.4, 1.6 ; in each case, run the algorithm on the same three start states and record the values of G and N in your table.

(d) Briefly discuss the tradeoff between speed and quality of solution for these five algorithms.

Question 3: Decision trees

Consider the decision tree learning algorithm of Figure 7.7 and the data of Figure 7.1 from Poole &Mackworth [1], also presented below. Suppose, for this question, the stopping criterion is that all of the examples have the same classification. The tree of Figure 7.6 was built by selecting a feature that gives the maximum information gain. This question considers what happens when a different feature is selected.

Example	*Author*	*Thread*	*Length*	*Where_read*	*User_action*
e_1	*known*	*new*	*long*	*home*	*skips*
e_2	*unknown*	*new*	*short*	*work*	*reads*
e_3	*unknown*	*followup*	*long*	*work*	*skips*
e_4	*known*	*followup*	*long*	*home*	*skips*
e_5	*known*	*new*	*short*	*home*	*reads*
e_6	*known*	*followup*	*long*	*work*	*skips*
e_7	*unknown*	*followup*	*short*	*work*	*skips*
e_8	*unknown*	*new*	*short*	*work*	*reads*
e_9	*known*	*followup*	*long*	*home*	*skips*
e_{10}	*known*	*new*	*long*	*work*	*skips*
e_{11}	*unknown*	*followup*	*short*	*home*	*skips*
e_{12}	*known*	*new*	*long*	*work*	*skips*
e_{13}	*known*	*followup*	*short*	*home*	*reads*
e_{14}	*known*	*new*	*short*	*work*	*reads*
e_{15}	*known*	*new*	*short*	*home*	*reads*
e_{16}	*known*	*followup*	*short*	*work*	*reads*
e_{17}	*known*	*new*	*short*	*home*	*reads*
e_{18}	*unknown*	*new*	*short*	*work*	*reads*
e_{19}	*unknown*	*new*	*long*	*work*	?
e_{20}	*unknown*	*followup*	*short*	*home*	?

Figure 7.1: Examples of a user's preferences

a) Suppose you change the algorithm to always select the first element of the list of features. What tree is found when the features are in the order [*Author, Thread, Length, WhereRead*]? Does this tree represent a different function than that found with the maximum information gain split? Explain.

b) What tree is found when the features are in the order [*WhereRead, Thread, Length, Author*]? Does this tree represent a different function than that found with the maximum information gain split or the one given for the preceding part? Explain.

c) Is there a tree that correctly classifies the training examples but represents a different function than those found by the preceding algorithms? If so, give it. If not, explain why.

Question 4: Decision trees

The goal is to take out-of-the-box models and apply them to a given dataset. The task is to analyse the data and build a model to predict whether income exceeds $50K/yr based on census data (also known as "Census Income" dataset).
Use the data set **Adult Data Set** from the Machine Learning repository [2].
Use the supervised learning methods discussed in the lectures, Decision Trees and Naive Bayes.

Do not code these methods: instead use the implementations from scikit-learn. Read the scikit-learn documentation on Decision Trees [3] and Naive Bayes [4], and the linked pages describing the parameters of the methods.

This question will help you master the workflow of model building. For example, you'll get to practice how to use the critical steps:

- Importing data
- Cleaning data
- Splitting it into train/test or cross-validation sets
- Pre-processing
- Transformations
- Feature engineering

Use the sklearn documentation pages for instructions. You should need the classification algorithms.

There are also available Tutorials:

- Sklearn – official tutorial for the sklearn package
- Predicting wine quality with scikit-learn – Step-by-step tutorial for training a machine learning model

The data is available here:

http://archive.ics.uci.edu/ml/machine-learning-databases/adult/

Preferences

1. Poole &Mackworth, Artificial Intelligence: Foundations of Computational Agents, Chapter 7, Supervised Machine Learning)

2. http://archive.ics.uci.edu/ml/datasets/Adult).

3. https://scikit-learn.org/stable/modules/tree.html

4. https://scikit-learn.org/stable/modules/naive_bayes.html

Submission

This assignment must be submitted electronically.
Put your zID and your name at the top of every page of your submission!

COMP3411 students should submit by typing

give cs3411 hw2 ...

(for example: give cs3411 hw2 assignment2.pdf)

COMP9814 students should submit by typing

give cs3411 hw2 ...

(for example: give cs9414 hw2 assignment2.pdf)

The give script will accept *.pdf *.txt *.doc *.rtf
Late submissions will incur a penalty of 10% per day, applied to the maximum mark.

Group submissions will not be allowed. By all means, discuss the assignment with your fellow students. But you must write (or type) your answers individually. **Do NOT copy anyone else's assignment, or send your assignment to any other student.**

Good luck!