CAB320 - 2023 Exam Prep Guide

Jaden Ussher

July 25, 2024

Question 1

Question: Calculate the edit distance between the two strings "sunda" and "satu" using dynamic programming. Show the table used in the computation and the final edit operations.

Solution

The table M is initialized as follows:

$$M = \begin{bmatrix} 0 & 1 & 2 & 3 & 4 \\ 1 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 0 \\ 5 & 0 & 0 & 0 & 0 \end{bmatrix}$$

The final M:

$$M = \begin{bmatrix} 0 & 1 & 2 & 3 & 4 \\ 1 & 0 & 1 & 2 & 3 \\ 2 & 1 & 1 & 2 & 2 \\ 3 & 2 & 2 & 2 & 3 \\ 4 & 3 & 3 & 3 & 3 \\ 5 & 4 & 3 & 4 & 4 \end{bmatrix}$$

The edit distance is therefore 4.

Edit Operations

$$\begin{array}{ccccc} s & u & n & d & a \\ s & - & a & t & u \end{array}$$

Reasoning

The solution utilizes the dynamic programming approach to compute the edit distance between two strings. The initialization sets up the base case where the transformation costs are directly related to the indices. The final matrix M provides the edit distance in the bottom-right cell, which is 4 in this case. The edit operations show the specific steps to transform one string into the other.

Question 2

Question: a) Is it guaranteed that A* tree search using h_2 will return an optimal solution? b) Suppose we are using a graph search version of A*, is it still guaranteed that it will return an optimal solution?

Solution

No, as h_2 is not necessarily admissible (for example, if $h_1 = h^*$), there is no guarantee that A* tree search will return an optimal solution.

Yes, it is. Let G_2 be the state graph obtained from the original state graph G_1 by multiplying each weight by 2. The function h_2 is admissible with G_2. For all nodes n in the search tree, $g_1(n) + h_2(n) \leq 2g_1(n) + h_2(n)$. This is true, particularly for the goal node n_g reached using A* with $2g_1 + h_2$ in G_2. This goal node n_g is also optimal in G_1 with respect to $g_1 + h_1$. The goal node n'_g reached with A* in G_1 with $g_1 + h_2$ satisfies $g_1(n'_g) + h_2(n'_g) \leq g_1(n_g) + h_2(n_g)$. Therefore, the cost of n'_g is at most twice the cost of an optimal solution of G_1.

Reasoning

The first part of the solution discusses the non-admissibility of h_2 if it were to always provide an overestimate, which contradicts the conditions for optimality in A* search. The second part of the solution uses a state graph transformation to argue that an admissible heuristic remains valid under certain transformations, ensuring that the cost of the goal node found remains within an acceptable bound.

Question 3

Question: Given the following inequalities, derive the bounds for the heuristic values $h(A)$, $h(S)$, $h(D)$, $h(E)$, and $h(C)$: - $h(A) \leq \min(5 + h(B), 1 + h(F))$ - $h(S) \leq 3 + h(A)$ - $\max(h(B) - 2, h(F) - 4) \leq h(D)$ - $h(D) \leq 3 + h(G)$ - $h(E) \leq \min(1 + h(F), 6 + h(G))$ - $h(C) \leq 2 + h(E)$

Solution

We have

$$h(A) \leq \min(5 + h(B), 1 + h(F)) = \min(7, 6) = 6$$

and

$$h(S) \leq 3 + h(A)$$

Therefore,

$$6 = 9 - 3 \leq h(A) \leq 6$$

We have

$$\max(h(B) - 2, h(F) - 4) \leq h(D) \quad \text{and} \quad h(D) \leq 3 + h(G)$$

Therefore

$$\max(0, 1) = 1 \leq h(D) \leq 3$$

We have

$$h(E) \leq \min(1 + h(F), 6 + h(G)) = \min(6, 6) = 6$$

and

$$h(C) \leq 2 + h(E)$$

Therefore,

$$5 = 7 - 2 \leq h(E) \leq 6$$

Reasoning

The solutions involve deriving bounds on heuristic values based on given inequalities. Each part systematically applies the given conditions to find the upper and lower bounds for the heuristic values of different nodes in the search graph.

Question 4

Question: Given the observation sequence $S, A, R, S, A, R, S = 3, 1, 1, 2, 2, 0, 4$, update the Q-values using Q-learning. Explain why the agent might fail to learn a good policy and suggest methods to ensure exploration.

Solution

The update for Q-learning is:

$$Q[s, a] = (1 - \text{lr}) \cdot Q[s, a] + \text{lr} \cdot (r + \text{df} \cdot \max(Q[\text{sn}, \text{an}] \text{ for an in actions(sn)}))$$

$$\begin{aligned}
&Q(3, 1) \text{ will be updated.}\\
&Q(3, 1) = (1 - 0.2) \cdot 0.3 + 0.2 \cdot (1 + 0.9 \cdot \max(0.2, 0.6, 1.0)) = 0.62\\
&Q(2, 2) \text{ will be updated.}\\
&Q(2, 2) = (1 - 0.2) \cdot 0.6 + 0.2 \cdot (0 + 0.9 \cdot \max(0.4, 0.8, 1.2)) = 0.696
\end{aligned}$$

The agent will not explore its action space and won't be able to discover better actions than the greedy actions of the current policy derived from Q. The agent will fail to learn a good policy.

The agent can explore its action space by selecting a random action 10% of the time. It can also use the UCB formula to guide its choice of actions.

Reasoning

The updates for Q-values follow the Q-learning update rule, incorporating learning rate (lr) and discount factor (df). The lack of exploration leads to suboptimal learning, and incorporating exploration mechanisms such as ϵ-greedy or UCB helps the agent discover better policies.

Question 5

Question: Given the following probabilities: - $P(C) = 1\%$ - $P(T|C) = 80\%$ - $P(T|\neg C) = 9.6\%$

Calculate the probability that Alice has cancer given that she tested positive.

Solution

Let's define some variables:

- C: has cancer
- T: test positive

From the text, we derive:

$$\begin{aligned} P(C) &= 1\%, & P(\neg C) &= 99\% \\ P(T|C) &= 80\%, & P(\neg T|C) &= 20\% \\ P(T|\neg C) &= 9.6\%, & P(\neg T|\neg C) &= 90.4\% \end{aligned}$$

In the case of Alice, we are interested in $P(C|T)$:

$$P(C|T) = \frac{P(T|C) \cdot P(C)}{P(T)} \quad \text{where} \quad P(T) = P(T, C) + P(T, \neg C) = P(T|C) \cdot P(C) + P(T|\neg C) \cdot P(\neg C)$$

Numerically,

$$\begin{aligned} P(T) &= 0.8 \cdot 0.01 + 0.096 \cdot 0.99 = 0.008 + 0.09504 \\ P(C|T) &= \frac{0.008}{0.008 + 0.09504} = 0.0776 \end{aligned}$$

That is, Alice has only about a 7.8% chance of having cancer given that the test result is positive.

Reasoning

The solution uses Bayes' theorem to calculate the posterior probability of having cancer given a positive test result. It accounts for the prior probabilities of having cancer and not having cancer, as well as the likelihood of testing positive in each case.

Question 6

Question: Calculate the posterior probability ratio $P(Y = 1|A = 0, B = 1, C = 0, D = 0)$ to $P(Y = 0|A = 0, B = 1, C = 0, D = 0)$ using the Naive Bayes hypothesis.

Solution

Using the Naive Bayes hypothesis:

$$P(Y|A, B, C, D) = \frac{P(A, B, C, D|Y) \cdot P(Y)}{P(A, B, C, D)}$$

We approximate $P(A, B, C, D|Y)$ with $P(A|Y) \cdot P(B|Y) \cdot P(C|Y) \cdot P(D|Y)$.

The ratio becomes:

$$\frac{P(Y = 1) \cdot P(A = 0|Y = 1) \cdot P(B = 1|Y = 1) \cdot P(C = 0|Y = 1) \cdot P(D = 0|Y = 1)}{P(Y = 0) \cdot P(A = 0|Y = 0) \cdot P(B = 1|Y = 0) \cdot P(C = 0|Y = 0) \cdot P(D = 0|Y = 0)}$$

Numerically:

$$\frac{(3/7) \cdot (2/3) \cdot (3/3) \cdot (2/3) \cdot (1/3)}{(4/7) \cdot (1/4) \cdot (2/4) \cdot (1/4) \cdot (3/4)} = 0.0634920$$

over

$$0.01339285$$

The ratio equals 4.74074.

Reasoning

The solution applies the Naive Bayes formula to calculate the posterior probability ratio given the likelihoods and prior probabilities. This approach simplifies the joint probability computation by assuming conditional independence among the features.

Question 7

Question: Calculate the information gain for attributes $F1$ and $F2$ given the entropy of the distribution (p, n) for both attributes. Which attribute should be chosen based on the highest information gain?

Solution

Let $H(p, n)$ denote the entropy of the distribution (p, n).

The information gain for $F1$ is:

$$H(3,3) - \left(\frac{3}{6} \cdot H(2,1) + \frac{3}{6} \cdot H(1,2)\right)$$

The information gain for $F2$ is:

$$H(3,3) - \left(\frac{2}{6} \cdot H(1,1) + \frac{2}{6} \cdot H(1,1) + \frac{2}{6} \cdot H(1,1)\right)$$

Using bits, with the following Python code:

```
from math import log2
H = lambda p, n : (-p/(p+n))*log2(p/(p+n)) + (-n/(p+n))*log2(n/(p+n))
```

We find that the information gain for $F1$ is:

$$H(3,3) - \left(\frac{3}{6} \cdot H(2,1) + \frac{3}{6} \cdot H(1,2)\right) = 0.0817041$$

The information gain for $F2$ is:

$$\frac{2}{6} \cdot H(1,1) + \frac{2}{6} \cdot H(1,1) + \frac{2}{6} \cdot H(1,1) - H(3,3) = 0$$

The attribute $F1$ should be chosen as it has the largest information gain.

Reasoning

The solution calculates the information gain for two features using entropy. The feature with the highest information gain is preferred for splitting the data in a decision tree, as it provides the most significant reduction in uncertainty.