COMP3032: Intelligent Algorithms Coursework

Thomas Smith

December 8, 2011

1 Abstract

This coursework involved using Hidden Markov models to analyse the dishonest casino problem. Given a specification of the transition and emission probabilities of two HMMs, we looked at applying various algorithms to determine specific information about the scenario. The two models were similar in most respects: they each involved two dice (distinct 'states'), and the chances of transitioning from one to the other were the same in both cases. Each model had one fair die, and one that was loaded, with the one in model 1 (M_1) favouring even numbers 5:1, and the one in model 2 (M_2) favouring odd numbers with the same ratio.

2 Tasks

Generate sequences from each model

In order to generate the sequences the models were set up with start probabilities based on the equilibrium distribution ($S_1 = \frac{1}{3}$, $S_2 = \frac{2}{3}$). Originally the coursework was developed in Octave, which provides a useful `discrete_rnd(c, v, p)` function for generating discrete values according to a specific probability distribution. This does not exist in the Matlab environment however, so for handin a functionally similar replacement was written. The `gensequence(loaded,T)` function that was developed also returns the state sequence that was produced while generating the dice rolls, which later helped in verifying the output of the `viterbi(seq,loaded)` function.

Calculate the probability of a sequence

After loading the provided sequences, the forward algorithm was used to calculate the likelihood that each sequence was generated by each model. Again, the algorithm was set up using the equilibrium distribution and the chance of the first observation, and then marched forwards through the sequence calculating the probability of either state given the emissions so far; finally summing these probabilities to give the probabilities listed in *Table 1*.

From these results, it can be seen that the chances of any of these specific sequences being produced by either HMM is extraordinarily small, though there is a noticable difference in likelihood between the results for each model. By looking at which HMM provides a greater probability, we can infer which model was used to create a specific sequence: i.e we can deduce that `my_roll_one` and `roll_one` were produced by M_1, and `my_roll_two`, `roll_two` & `roll_three` were produced by M_2. In general, the probabilities between the models differ by 10-15 orders of magnitude, however for `roll_one` the difference in probabilities is about half that size, meaning that is is closer to a sequence that could have been generated by a single fair die.

Table 1	M_1	M_2
`my_roll_one`	7.59×10^{-71}	3.65×10^{-86}
`my_roll_two`	6.85×10^{-85}	9.30×10^{-75}
`roll_one`	1.74×10^{-77}	1.98×10^{-83}
`roll_two`	1.54×10^{-86}	6.61×10^{-72}
`roll_three`	5.80×10^{-86}	4.85×10^{-71}

Estimate hidden state sequence

For each sequence and each model, the Viterbi algorithm was used to estimate the most likely sequence of hidden states resulting in the observed emissions. The product of the equilibrium distribution and the chance of the first observation provided the starting point for the algorithm, and then at each timestep the most probable state was stored, along with the individual probabilities at the current step, which were used for the next calculation. In order to minimize the risk of errors due to floating-point underflow, the logarithms of the probabilities were used, and the algorithm altered accordingly. When compared to the actual states used for `my_roll_one`, the Viterbi algorithm produces a reasonable estimation.