Assignment4 – Markov Models

Given: May 23 Due: May 30

Problem 4.1 (Prediction, Filtering, Smoothing)

Consider an HMM consisting of a stationary Markov process and a stationary sensor model. We restrict attention to HMMs with a single state-variable X_t and a single evidence variable E_t. (The sensor model does not necessarily have the Markov property.)

Explain the results of the *prediction*, *filtering*, *smoothing* algorithms. For each one, state the motivation, the expression for the conditional probability that is to be computed, and explain the components of the formula. You do not have to explain how the algorithms work.

Solution: In all three cases, we have observed the process from time 1 to time t, i.e., we have observed the events $E_i = e_i$ for $i = 1, \ldots, t$. Let $E_{1:t} = e_{1:t}$ abbreviate the conjunction of these events.

We use those observations to compute different conditional probabilities about the state X_p of the HMM at time p.

Prediction Here we predict the future of the HMM, i.e., $p = t + k > t$. We need the conditional probability

$$P(X_{t+k} \mid E_{1:t} = e_{1:t})$$

Filtering Here we estimate the current state of the HMM, i.e., $p = t$. We need the conditional probability

$$P(X_t \mid E_{1:t} = e_{1:t})$$

Smoothing Here we estimate the past states of the HMM, i.e., $p = k < t$. We need the conditional probability

$$P(X_k \mid E_{1:t} = e_{1:t})$$

Problem 4.2 (HMMs in Python)

Implement filtering, *prediction* and *smoothing* for *HMMs* in Python by completing the *implementation* of `hmm.py` at `https://kwarc.info/teaching/AI/resources/AI2/hmm/`.

Hint: This problem uses `numpy`, which is a Python *library* for working with arrays/matrices. If you have never worked with `numpy` before, you can find many high-quality introductions online. Due to its popularity and frequent use for machine learning etc., it is definitely worth getting to know `numpy`. That being said,

you only need very few and basic numpy functions for this assignment, which you should be able to find without problems (e.g. searching for *numpy matrix multiplication*).

Solution: See `https://kwarc.info/teaching/AI/resources/AI2/hmm/`.
