CS B551, Fall 2015, Statistical inference and learning practice problems

1. A certain little-known human language consists of only nouns and verbs. A sentence begins with a noun with probability 75% and with a verb with probability 25%. If a given word in a sentence is a noun, there is a 75% probability that the next word is a verb and a 25% probability that the next word is a noun. If a given word is a verb, there is a 75% probability that the next word in the sentence is a noun, and a 25% probability that is a verb.

   We can model this language using a Markov chain with two states: noun and verb.

   (a) *(3 pts)* Draw the Markov chain described above.

   (b) *(3 pts)* What is the probability that the first three words of a sentence are *all* nouns?

   $$(0.75)(0.25)(0.25) \approx 0.0469$$

   (c) *(4 pts)* What is the probability that the third word of a sentence is a noun?
   For the third word to be a noun, the first three words must be NNN, NVN, VNN, or VVN. Thus the probability that the third word is a noun is:

   $$(0.75)(0.25)(0.25) + (0.75)(0.75)(0.75) + (0.25)(0.75)(0.25) + (0.25)(0.25)(0.75) = .5625$$

   (d) There are only four words in the language: `awk`, `yacc`, `grep`, and `perl`. Each of these words can be either a noun or a verb depending on context. Of all noun occurrences, 10% are the word `awk`, 20% are the word `yacc`, 40% are the word `grep`, and 30% are the word `perl`. Of all verb occurrences, 20% are the word `awk`, 30% are the word `yacc`, 45% are the word `grep`, and 5% are the word `perl`. (*Hint:* These are the emission probabilities of the HMM; for example, $e_{\text{noun}}(\texttt{yacc}) = 0.2$.)

   (10 pts) Using this model and the Viterbi algorithm, find the most likely part-of-speech state sequence for the following sentence: `Perl yacc awk awk.`
   Using Viterbi, we find that the best sequence is noun verb noun verb.

2. A particularly ruthless instructor gives unannounced quizzes and unannounced exams. On any given day of his class, there is either a quiz (Q), an exam (E), or a lecture (L). The sequence of classes for Fall 2007 was like this:

   LLLQLLQLQLLLQLELLLLLQLLQQLLLQE

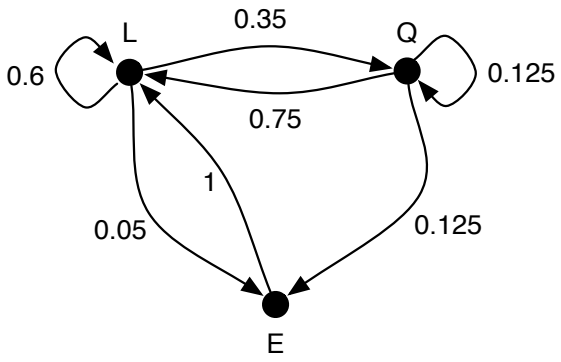
   In other words, there were quizzes on days 4, 7, 9, 13, 21, 24, 25, and 29, exams on days 15 and 30, and lectures on the other days. We can construct a Markov chain to model the instructor's behavior.

(a) Estimate $P(X_{n+1} = Q|X_n = L)$, the probability of having a quiz the day after a lecture, using the observed sequence of classes above.

There were 20 days with lectures, and for seven of those 20 days there was a quiz the next day, so the probability is $7/20 = 35\%$.

(b) Estimate the other transition probabilities, and draw the Markov chain.

$$\begin{aligned}
P(X_{n+1} = Q|X_n = L) &= 7/20 = 0.35 \\
P(X_{n+1} = E|X_n = L) &= 1/20 = 0.05 \\
P(X_{n+1} = L|X_n = L) &= 12/20 = 0.6 \\
P(X_{n+1} = Q|X_n = Q) &= 1/8 = 0.125 \\
P(X_{n+1} = E|X_n = Q) &= 1/8 = 0.125 \\
P(X_{n+1} = L|X_n = Q) &= 6/8 = 0.75 \\
P(X_{n+1} = Q|X_n = E) &= 0/1 = 0 \\
P(X_{n+1} = E|X_n = E) &= 0/1 = 0 \\
P(X_{n+1} = L|X_n = E) &= 1/1 = 1
\end{aligned}$$



(c) The model of (b) can be used to predict the instructor's behavior in future semesters. Suppose that the first class of the semester is always a lecture, and the remaining sequence of classes are chosen according to the Markov chain of part (b).

i. What is the probability that the first five classes of next semester will be LLLQE?
$(1.0)(0.6)(0.6)(0.35)(0.125) = 63/4000 = 0.01575$

ii. What is the probability that the first five classes of next semester will be LLLEQ?
$(1.0)(0.6)(0.6)(0.05)(0) = 0$

iii. What is the probability that the first five classes of next semester will be LQQQQ?
$(1.0)(0.35)(0.125)^3 \approx .000684$

(d) Give two disadvantages of using a Markov chain to model the instructor's behavior. (That is, give two constraints that the instructor is probably using to decide when to schedule quizzes and exams, but that the Markov model cannot capture.)

There are many possible answers; here are a few:

- The instructor would give more lectures than quizzes or exams, but the probability of a sequence like LQQQQQQQ... is non-zero according to the Markov chain.
- The instructor would also give a least one exam or quiz, but the probability of the sequence LLLLL.... is non-zero according to the Markov chain.
- The instructor would give exams at regular intervals (for example, one near the middle of the semester and one at the end), but there is no way for the Markov chain to capture this.

3. What is the expected number of times that a fair six-sided die must be rolled before six 6's occur in a row?

   (*Hint:* think about this problem in terms of a Markov Chain. In particular, start by building a Markov Chain with seven states, labeled 0 through 6. The state of the Markov model records the number of consecutive 6's that have been rolled. To answer the question, find the expected number of time steps required until the machine enters state 6 for the first time.)

   Let X_n be a random variable denoting the number of rolls required to obtain n consecutive 6's. The number of rolls required to obtain zero consecutive 0's is always 0, so $E[X_0] = 0$.

   Now consider the value of $E[X_n]$, for $n \geq 1$. To enter state n the system must be in state $n-1$. The expected number of rolls to reach state $n-1$ can be written (recursively) as $E[X_{n-1}]$. If the system is currently in state $n-1$, then one of two transitions occurs next:

   (a) with probability $\frac{1}{6}$, the system enters state n. In this case the expected number of steps to reach state n is just $E[X_{n-1}] + 1$.

   (b) with probability $\frac{5}{6}$, the system returns to state 0. In this case the expected number of steps to reach state n can be written recursively as $E[X_{n-1}] + 1 + E[X_n]$. (Intuitively, $E[X_{n-1}] + 1$ steps were just "wasted" because we returned to state 0. So the number of steps to reach n is the number of wasted time steps, plus the expected number of time steps it will take to reach state n.)

   Taken together, these two possible transitions give,

$$E[X_n] = (\frac{1}{6})(E[X_{n-1}] + 1) + (\frac{5}{6})(E[X_{n-1}] + 1 + E[X_n]),$$

   which simplifies to,

$$E[X_n] = \frac{E[X_{n-1}] + 1}{p}.$$

   Using this equation and the fact that $E[X_0] = 0$ gives the answer, $E[X_6] = 55,986$ rolls.

4. A meteorological observing station in Ithaca uses a wireless network link to transmit data from a moisture sensor to a central computer. Each day the sensor sends a `yes` message to the computer if it has rained that day, or a `no` message if it has not rained. Unfortunately,

the wireless link is noisy, so that 40% of the `yes` messages sent by the sensor are incorrectly received as `no` messages by the computer, and 20% of the `no` messages transmitted by the sensor are incorrectly received as `yes` messages by the computer. Suppose that the weather in Ithaca can be modeled by a simple Markov chain: if it rains one day, the probability of it raining the next day is 65%; if it does not rain one day, the probability of rain the next day is 25%.

During the first week of operation, the computer receives the following sequence of messages from the sensor:

```
yes yes no yes no no yes
```

Use the Viterbi algorithm to estimate whether or not it was actually raining on each day of the week. Assume that on the first day of the week, there was a 50% chance of rain in Ithaca.

This problem can be modeled as an HMM with the following parameters:

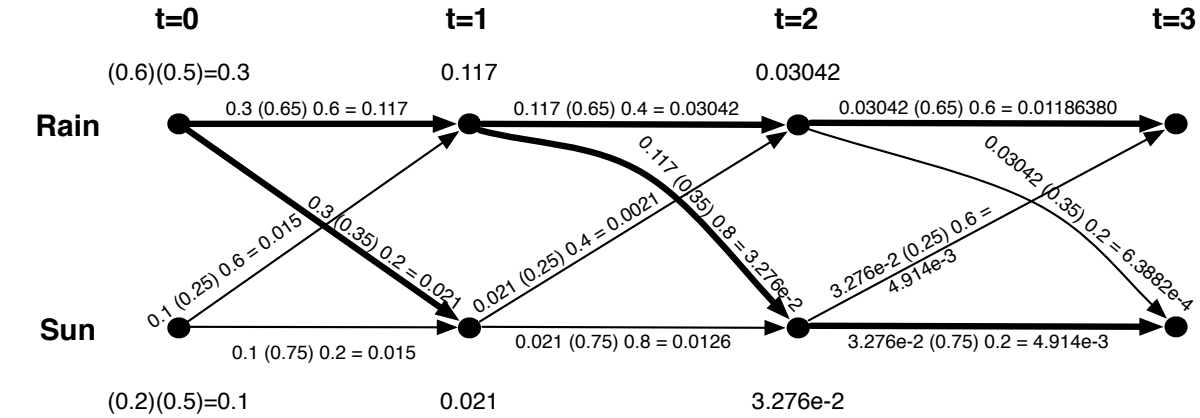
- **Initial distribution:** $P(X_0 = R) = 0.5$, $P(X_0 = S) = 0.5$ (where R and S represent rain and sun, respectively).
- **Transition probabilities:**

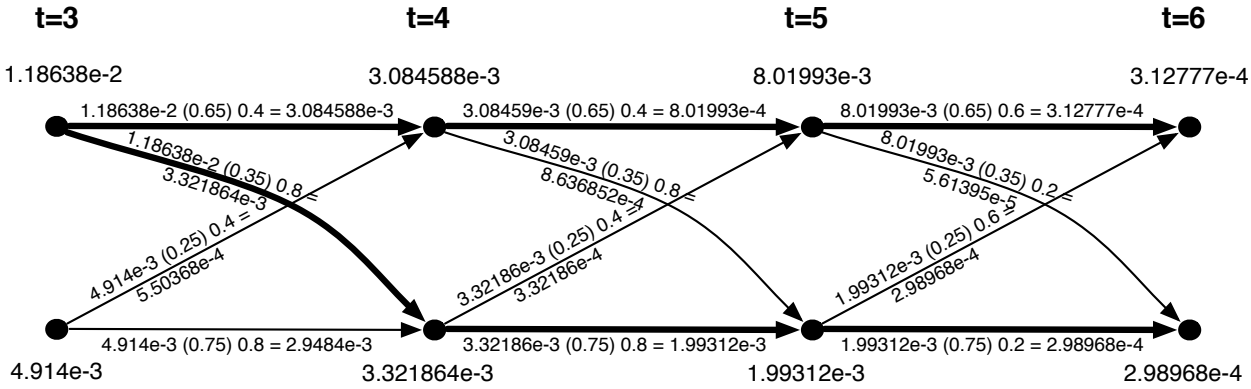
$$P(X_{t+1} = R|X_t = R) = 0.65$$
$$P(X_{t+1} = S|X_t = R) = 0.35$$
$$P(X_{t+1} = R|X_t = S) = 0.25$$
$$P(X_{t+1} = S|X_t = S) = 0.75$$

- **Emission probabilities:**

$$E_R(Y) = 0.6, E_R(N) = 0.4$$
$$E_S(Y) = 0.2, E_S(N) = 0.8$$

Then the Viterbi algorithm can be used to find the most likely state sequence:





From the result at $t = 6$, we see that the most likely state sequence ends with rain. Following the bolded arrows backwards, we find the mostly likely sequence: RRRRRRR.

5. *(2 pts)* Suppose that we're given a set of observations $D = \{x_1, x_2, ..., x_n\}$, with $x_i \in \mathcal{R}$, sampled from a Gaussian distribution with parameters $\mu = 0$ and an unknown σ. Recall that to perform Bayesian parameter estimation, we begin by constructing a meta-network that includes the unknown parameters of the Gaussian as random variables, and models the fact that each x_i is independent from the others conditioned on these unknown parameters.

   (a) Draw the meta-network for the case that $n = 3$.
   The Bayes net has a root note σ, with nodes x_1, x_2, ... x_n as children.

   (b) Suppose that $P(\sigma)$ is a uniform distribution. Draw a plot of the distribution $P(\sigma|x_1, x_2)$. Then draw a plot of the distribution $P(\sigma|x_1, ..., x_{20})$.

   (c) Explain how to calculate the distribution $P(x_{n+1}|x_1, ..., x_n)$.
   Just use variable elimination.

6. Most people with colorblindness are able to perceive color, but occasionally confuse certain colors with one another. Suppose that George can correctly identify red and green 75% of the time, but confuses the two colors 25% of the time. (For example, when presented with

a red object, he sees it as red 75% of the time but as green 25% of the time.) His son Erik, who is also red-green colorblind, correctly identifies colors 90% of the time. The traffic light is red 30% of the time and green 70% of the time.

(a) Draw the Bayes Net for this situation in terms of three random variables, P, G, and L (what Erik reports, what George reports, and what the light is).
Simple model with L as root, and E and G as children of L.

(b) Explain how to sample a single particle from the posterior, $P(E, G, L)$.
Use forward sampling, i.e., first choose L by flipping a biased coin, then choose G and L by flipping biased coins based on sampled value of L. For example, if you've chosen red for L, then flip a coing for G where the probability of $G = red$ is 0.75.

(c) Write some Python code to sample from $P(G, L|E = red)$ using MCMC, and to estimate from the samples $P(G = green|E = red)$ and $P(L = green|E = red)$.

```
import random

# initial sample -- assigned arbitrarily
# we're using 0="red", 1="green" here
(E, G, L) = (0, 0, 0)
samples = []
for n in range(0, 10000):
    # first sample a new value of G given L and E
    G = L if random.random() < 0.75 else 1-L

    # now sample from P(L | E,G) given E and G.
    # First figure out this distribution:
    #   P_red = P(L=red, E, G)/P(E,G)
    #   P_green = P(L=green, E, G) / P(P,G)
    L_dist = ( 0.3 * (0.75 if 0 == G else 0.25) * (0.9 if 0 == E else 0.1),
               0.7 * (0.75 if 1 == G else 0.25) * (0.9 if 1 == E else 0.1) )

    # instead of actually figuring out P(P,G), just normalize P_red and P_green so that they sum to 1
    L = 0 if (random.random() < L_dist[0] / sum(L_dist) ) else 1

    samples += [(E, G, L),]
    print (E, G, L)

print samples
G_green_count = sum( [ g for (e, g, l) in samples ] )
L_green_count = sum( [ l for (e, g, l) in samples ] )
print "P(G=green | E=red) = " + str(G_green_count / float(len(samples))) + \
    ", actual: " + str( (0.3 * 0.25 * 0.9 + 0.7 * 0.75 * 0.1) / ( 0.7 * 0.1 + 0.3 * 0.9 ) )
print "estimated P(L=green | E=red) = " + str(L_green_count / float(len(samples))) + \
    ", actual: " + str( 0.1*0.7 / (0.1*0.7 + 0.9*0.3) )
```