CSE 250A. Assignment 1

Out: *Tue Sep 29*
Due: *Tue Oct 06* (end of class — **no extensions**)
Reading: Russell & Norvig, Chapter 13.

1.1 Conditioning on background evidence [RN 13.16]

It is often useful to consider the impact of specific events in the context of general background evidence, rather than in the absence of information.

(a) Denoting such evidence by E, prove the conditionalized version of the product rule:

$$P(X, Y|E) \;=\; P(X|Y, E)P(Y|E).$$

(b) Also, prove the conditionalized version of Bayes rule:

$$P(X|Y, E) \;=\; \frac{P(Y|X, E)P(X|E)}{P(Y|E)}.$$

1.2 Conditional independence [RN 13.17]

Show that the following three statements about random variables X, Y, and E are equivalent:

$$\begin{aligned} P(X, Y|E) &= P(X|E)P(Y|E) \\ P(X|Y, E) &= P(X|E) \\ P(Y|X, E) &= P(Y|E) \end{aligned}$$

You should become fluent with all these ways of expressing that X is conditionally independent of Y given E.

1.3 Creative writing

Attach events to the binary random variables X, Y, and Z that are consistent with the following patterns of commonsense reasoning. You may use different events for the different parts of the problem.

(a) Cumulative evidence:

$$P(X=1) < P(X=1|Y=1) < P(X=1|Y=1, Z=1)$$

(b) Explaining away:

$$\begin{aligned} P(X=1|Y=1) &> P(X=1), \\ P(X=1|Y=1, Z=1) &< P(X=1|Y=1) \end{aligned}$$

(c) Conditional independence:

$$\begin{aligned} P(X=1, Y=1) &\neq P(X=1)P(Y=1) \\ P(X=1, Y=1|Z=1) &= P(X=1|Z=1)P(Y=1|Z=1) \end{aligned}$$

1.4 Bayes Rule

Suppose that 1% of competitive cyclists use performance-enhancing drugs and that a particular drug test has a 5% false positive rate and a 10% false negative rate.

(a) Cyclist A tests negative for drug use. What is the probability that Cyclist A is not using drugs?

(b) Cyclist B tests positive for drug use. What is the probability that Cyclist B is using drugs?

1.5 Entropy

(a) Let X be a discrete random variable with $P(X = x_i) = p_i$ for $i \in \{1, 2, \ldots, n\}$. The entropy $\mathcal{H}[X]$ of the random variable X is a measure of its uncertainty. It is defined as:

$$\mathcal{H}[X] \;=\; -\sum_{i=1}^{n} p_i \log p_i.$$

Show that the entropy $\mathcal{H}[X]$ is maximized when $p_i = \frac{1}{n}$ for all i. You should do this by computing the gradient with respect to p_i and using Lagrange multipliers to enforce the constraint that $\sum_i p_i = 1$. Later in the course, we will use similar calculations for learning probabilistic models.

(b) The joint entropy of n discrete random variables $(X_1, X_2, \ldots, X_n)$ is defined as:

$$H(X_1, X_2, \ldots, X_n) \;=\; -\sum_{x_1}\sum_{x_2}\cdots\sum_{x_n} P(x_1, x_2, \ldots, x_n) \log P(x_1, x_2, \ldots, x_n),$$

where the sums range over all possible instantiations of $X_1, X_2, \ldots, X_n$. Show that if the variables X_i are independent, then their joint entropy is the sum of their individual entropies: namely,

$$P(x_1, x_2, \ldots, x_n) \;=\; \prod_{i=1}^{n} P(x_i) \quad \text{implies} \quad H(X_1, X_2, \ldots, X_n) \;=\; \sum_{i=1}^{n} H(X_i).$$

1.6 Kullback-Leibler distance

Consider two discrete probability distributions, p_i and q_i, with $\sum p_i = \sum q_i = 1$. The Kullback-Leibler (KL) distance between these distributions (also known as the relative entropy) is defined as:

$$\mathrm{KL}(p, q) = \sum_i p_i \log(p_i/q_i).$$

(a) Consider the natural logarithm (in base e). By sketching graphs of $\log(x)$ and $x - 1$, verify the inequality:

$$\log(x) \;\leq\; x - 1,$$

with equality if and only if $x = 1$. Confirm this result by differentiation of $\log(x) - (x - 1)$.

(b) Use the previous result to prove that $\mathrm{KL}(p, q) \geq 0$, with equality if and only if the two distributions p_i and q_i are equal.

(c) Using the inequality in (a), as well as the simple equality $\log x = 2 \log \sqrt{x}$, derive the tighter lower bound:

$$\mathrm{KL}(p, q) \geq \sum_i (\sqrt{p_i} - \sqrt{q_i})^2.$$

(d) Provide a counterexample to show that the KL distance is not a symmetric function of its arguments:

$$\mathrm{KL}(p, q) \neq \mathrm{KL}(q, p).$$

Despite this asymmetry, it is still common to refer to $\mathrm{KL}(p, q)$ as a measure of distance. Many algorithms for machine learning are based on minimizing KL distances between probability distributions.

1.7 Hangman

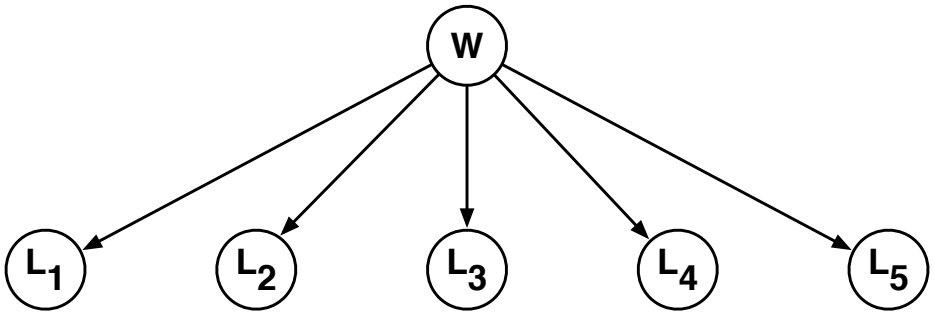
Consider the belief network shown below, where the random variable W stores a five-letter word and the random variable $L_i \in \{\mathtt{A}, \mathtt{B}, \ldots, \mathtt{Z}\}$ reveals only the word's ith letter. Also, suppose that these five-letter words are chosen at random from a large corpus of text according to their frequency:

$$P(W = w) = \frac{\textsc{count}(w)}{\sum_{w'} \textsc{count}(w')},$$

where $\textsc{count}(w)$ denotes the number of times that w appears in the corpus and where the denominator is a sum over all five-letter words. Note that in this model the conditional probability tables for the random variables L_i are particularly simple:

$$P(L_i = \ell | W = w) = \begin{cases} 1 & \text{if } \ell \text{ is the } i\text{th letter of } w, \\ 0 & \text{otherwise.} \end{cases}$$

Now imagine a game in which you are asked to guess the word w one letter at a time. The rules of this game are as follows: after each letter ($\mathtt{A}$ through $\mathtt{Z}$) that you guess, you'll be told whether the letter appears in the word and also where it appears. Given the *evidence* that you have at any stage in this game, the critical question is what letter to guess next.



Let's work an example. Suppose that after three guesses—the letters D, I, M—you've learned that the letter I does *not* appear, and that the letters D and M appear as follows:

M ___ D ___ M

Now consider your next guess: call it ℓ. In this game the best guess is the letter ℓ that maximizes

$$P\Big(L_2 = \ell \textbf{ or } L_4 = \ell \;\Big|\; L_1 = \mathtt{M}, L_3 = \mathtt{D}, L_5 = \mathtt{M}, L_2 \notin \{\mathtt{D}, \mathtt{I}, \mathtt{M}\}, L_4 \notin \{\mathtt{D}, \mathtt{I}, \mathtt{M}\}\Big).$$

In other works, pick the letter ℓ that is most likely to appear in the blank (unguessed) spaces of the word. For any letter ℓ we can compute this probability as follows:

$$\begin{aligned}
&P\Big(L_2 = \ell \textbf{ or } L_4 = \ell \;\Big|\; L_1 = \mathtt{M}, L_3 = \mathtt{D}, L_5 = \mathtt{M}, L_2 \notin \{\mathtt{D}, \mathtt{I}, \mathtt{M}\}, L_4 \notin \{\mathtt{D}, \mathtt{I}, \mathtt{M}\}\Big) \\
&= \sum_w P\Big(W = w, L_2 = \ell \textbf{ or } L_4 = \ell \;\Big|\; L_1 = \mathtt{M}, L_3 = \mathtt{D}, L_5 = \mathtt{M}, L_2 \notin \{\mathtt{D}, \mathtt{I}, \mathtt{M}\}, L_4 \notin \{\mathtt{D}, \mathtt{I}, \mathtt{M}\}\Big), \quad \boxed{\textbf{marginalization}} \\
&= \sum_w P(W = w | L_1 = \mathtt{M}, L_3 = \mathtt{D}, L_5 = \mathtt{M}, L_2 \notin \{\mathtt{D}, \mathtt{I}, \mathtt{M}\}, L_4 \notin \{\mathtt{D}, \mathtt{I}, \mathtt{M}\}\Big)\, P(L_2 = \ell \textbf{ or } L_4 = \ell | W = w) \quad \boxed{\textbf{product rule \& CI}}
\end{aligned}$$

where in the third line we have exploited the conditional independence (**CI**) of the letters L_i given the word W. Inside this sum there are two terms, and they are both easy to compute. In particular, the second term is more or less trivial:

$$P(L_2 = \ell \textbf{ or } L_4 = \ell | W = w) \;=\; \begin{cases} 1 & \text{if } \ell \text{ is the second or fourth letter of } w \\ 0 & \text{otherwise.} \end{cases}$$

And the first term we obtain from Bayes rule:

$$\begin{aligned}
&P(W = w | L_1 = \mathtt{M}, L_3 = \mathtt{D}, L_5 = \mathtt{M}, L_2 \notin \{\mathtt{D}, \mathtt{I}, \mathtt{M}\}, L_4 \notin \{\mathtt{D}, \mathtt{I}, \mathtt{M}\}\Big) \\
&= \frac{P(L_1 = \mathtt{M}, L_3 = \mathtt{D}, L_5 = \mathtt{M}, L_2 \notin \{\mathtt{D}, \mathtt{I}, \mathtt{M}\}, L_4 \notin \{\mathtt{D}, \mathtt{I}, \mathtt{M}\} | W = w\Big) P(W = w)}{P(L_1 = \mathtt{M}, L_3 = \mathtt{D}, L_5 = \mathtt{M}, L_2 \notin \{\mathtt{D}, \mathtt{I}, \mathtt{M}\}, L_4 \notin \{\mathtt{D}, \mathtt{I}, \mathtt{M}\})} \quad \boxed{\textbf{Bayes rule}}
\end{aligned}$$

In the numerator of Bayes rule are two terms; the left term is equal to zero or one (depending on whether the evidence is compatible with the word w), and the right term is the prior probability $P(W = w)$, as determined by the empirical word frequencies. The denominator of Bayes rule is given by:

$$\begin{aligned}
&P(L_1 = \mathtt{M}, L_3 = \mathtt{D}, L_5 = \mathtt{M}, L_2 \notin \{\mathtt{D}, \mathtt{I}, \mathtt{M}\}, L_4 \notin \{\mathtt{D}, \mathtt{I}, \mathtt{M}\}) \\
&= \sum_w P(W = w, L_1 = \mathtt{M}, L_3 = \mathtt{D}, L_5 = \mathtt{M}, L_2 \notin \{\mathtt{D}, \mathtt{I}, \mathtt{M}\}, L_4 \notin \{\mathtt{D}, \mathtt{I}, \mathtt{M}\}), \quad \boxed{\textbf{marginalization}} \\
&= \sum_w P(W = w) P(L_1 = \mathtt{M}, L_3 = \mathtt{D}, L_5 = \mathtt{M}, L_2 \notin \{\mathtt{D}, \mathtt{I}, \mathtt{M}\}, L_4 \notin \{\mathtt{D}, \mathtt{I}, \mathtt{M}\} | W = w), \quad \boxed{\textbf{product rule}}
\end{aligned}$$

where again all the right terms inside the sum are equal to zero or one. Note that the denominator merely sums the empirical frequencies of words that are compatible with the observed evidence.

Now let's consider the general problem. Let E denote the evidence at some intermediate round of the game: in general, some letters will have been guessed correctly and their places revealed in the word, while

other letters will have been guessed incorrectly and thus revealed to be absent. There are two essential computations. The first is the *posterior* probability, obtained from Bayes rule:

$$P(W=w|E) \;=\; \frac{P(E|W=w)\,P(W=w)}{\sum_{w'} P(E|W=w')\,P(W=w')}.$$

The second key computation is the *predictive* probability, based on the evidence, that the letter ℓ appears somewhere in the word:

$$P\Big(L_i=\ell \text{ for some } i\in\{1,2,3,4,5\}\Big|E\Big) \;=\; \sum_w P\Big(L_i=\ell \text{ for some } i\in\{1,2,3,4,5\}\Big|W=w\Big)P\Big(W=w\Big|E\Big).$$

Note in particular how the first computation feeds into the second. Your assignment in this problem is implement both of these calculations. **You may program in the language of your choice.**

(a) Download the file *hw1_word_counts_05.txt* that appears with the homework assignment. The file contains a list of 5-letter words (including names and proper nouns) and their counts from a large corpus of Wall Street Journal articles (roughly three million sentences). From the counts in this file compute the prior probability $P(w) = \text{COUNT}(w)/\text{COUNT}_{\text{total}}$. **As a sanity check, print out the eight most frequent 5-letter words, as well as the eight least frequent 5-letter words. Do your results make sense?**

(b) Consider the following stages of the game. For each of the following, indicate the best next guess—namely, the letter ℓ that is most likely (probable) to be among the missing letters. Also report the probability $P(L_i=\ell \text{ for some } i\in\{1,2,3,4,5\}|E)$ for your guess ℓ. Your answers should fill in the last two columns of this table.

correctly guessed	incorrectly guessed	best next guess ℓ	$P(L_i=\ell \text{ for some } i\in\{1,2,3,4,5\}\|E)$
– – – – –	{}		
– – – – –	{E,O}		
D – – I –	{}		
D – – I –	{A}		
– U – – –	{A,I,E,O,S}		

(c) Turn in a **hard-copy printout** of your source code. Do not forget the source code: it is worth many points on this assignment.

Checking your work: The demo on Piazza (also under *resources*) implements this program for words of length 6-10. You will also find count files for words of these lengths on Piazza, and if you modify your code to handle these different word lengths, you will be able to check your answers against the demo. (This is totally optional, though.) Just to be perfectly clear, you are **not** required in this problem to implement a user interface or any general functionality for the game of hangman. You will only be graded on your word lists in (a), the completed table for (b), and your source code in (c).