

Fall 2008 – COMP 131 Midterm 2 Review – Solutions¹

1 Bayes Nets

Let's say you go to the horse races. A shady character comes to offer you a free tip (a tip is a piece of information): he says Belle did not eat her breakfast. Assume that:

- The probability that a horse will win is dependent on the horses health and its speed.
- A horses health and its speed are independent.
- A healthy horse has a higher probability of eating breakfast than does a sick horse.
- Your informant is known to be accurate 80% of the time.

1. Draw a Bayesian network with 5 variables (T = you got this tip; B = Belle ate her breakfast; H = Belle is healthy; W = Belle will win; F = Belle is fast). The relationships between the variables should reflect the problem description.

Answer:

Nodes F and H have no parents.

Node W has two parents: H and F.

Node B has one parent: H

Node T has one parent: B

2. What is $P(W)$ (in terms of values stored in the networks conditional probability tables)?

Answer:

$$P(W) = \sum_{f,h} P(W|f,h)P(f)P(h)$$

3. What is $P(W|T)$ (in terms of values stored in the network's conditional probability tables)?

Answer:

$$\begin{aligned} P(W|T) &= \frac{P(W,T)}{P(T)} \\ &= \frac{\sum_{b,h,f} P(W|h,f)P(h)P(f)P(T|b)P(b|h)}{\sum_{b,h} P(T|b)P(b|h)P(h)} \end{aligned}$$

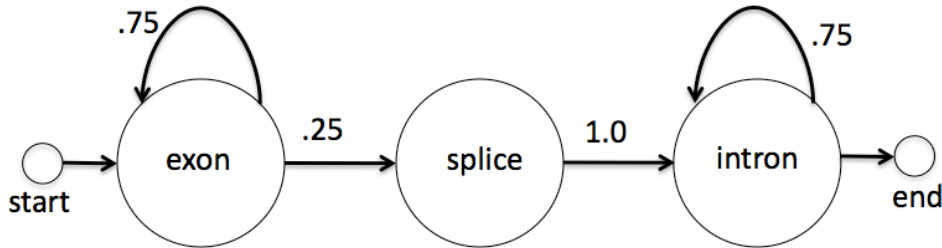
¹Based in part on MIT's OpenCourseWare materials for 6.034 Artificial Intelligence by Profs. Kaelbling and Lozano-Perez, Berkeley's AI course by Prof. Russell, and Rochester University CSC 242 by Prof. Brown.

2 Gene sequencing

A gene is a sequence of nucleotide bases A,T,C and G. Each gene is made of exons – sequences of bases which get transcribed into proteins, and introns – sequences that are not transcribed. The single nucleotide where the switch from one to the other occurs is called a splice site. In 95% of the cases, the splice site is a G. It can also be an A, rarely, but never a C or a T. All bases are equally likely to appear in an exon, but introns are A and T-heavy (A and T are 4 times as likely as C and G, and equally likely as each other). We expect quite a few bases in an exon or an intron sequence. Say, we know that if one nucleotide is part of an exon (or an intron), the next one in the sequence has a 75% chance of being part of the same exon (or intron). Assume that once a transition from exon to intron (through a splice site) has occurred, there will be no transitioning back.

1. Draw a graphical representation of our knowledge in this domain. What are the random variables in your model? Include any relevant probability distributions.

Answer: The random variables are the state X_i with values taken from the domain $\{exon, splice, intron\}$ and the evidence (base, nucleotide) E_i with values taken from the domain $\{a, t, c, g\}$ (lower-cased to underscore that these are discrete values of a random variable), for each place i in a sequence. The graphical model is shown below.



$P(E_i X_i)$			
a	.25	.05	.4
c	.25	0	.1
t	.25	0	.4
g	.25	.95	.1

Figure 1: Compact representation of the splice site HMM. A full representation with X_i and E_i for each base position i is also acceptable.

2. What kind of model have you drawn? Be specific.

Answer: This is a dynamic Bayes net. Because there is only one hidden state variable X , it is a hidden Markov model (HMM).

3. We have isolated a sequence CTGGAACTC, and we know that the first C comes from an exon, and the last C comes from an intron. Where is the most likely splice site between the two?

Answer: We are essentially asked for the most likely sequence of states that could have produced this sequence of observed evidence (bases). One way to compute that is with the Viterbi algorithm. The algorithm propagates forward a message, which at every time t is:

$$\begin{aligned}
\mathbf{m}_{1:t} &= \max_{x_1 \dots x_{t-1}} P(x_1 \dots x_{t-1}, X_t | e_1 \dots e_t) \\
&= \alpha P(e_t | X_t) \max_{x_{t-1}} (P(X_t | x_{t-1}) \mathbf{m}_{1:t-1})
\end{aligned}$$

In our case, given the evidence sequence *ctggaactc*, the algorithm will compute the following messages (in parantheses at the end, the most likely state at $t - 1$ is given, taken from the transition probabilities multiplied by message at time $t - 1$)²:

$$\begin{aligned}
\mathbf{m}_{1:1} &= \langle 1, 0, 0 \rangle, \text{ (start)} \\
\mathbf{m}_{1:2} &= \alpha P(E_2 = t | X_2) \max_{x_1} (P(X_2 | x_1) \mathbf{m}_{1:1}) \\
&\quad \begin{array}{c} X_2 = \text{exon} \quad \text{splice} \quad \text{intron} \\ X_1 = \text{exon} \begin{pmatrix} .75 & .25 & 0 \\ \text{splice} & 0 & 0 & 1 \\ \text{intron} & 0 & 0 & .75 \end{pmatrix} \times \langle 1, 0, 0 \rangle \end{array} \\
&= \alpha \langle .25, 0, .4 \rangle \langle .75, .25, 0 \rangle = \alpha \langle .1875, 0, 0 \rangle \text{ (exon, exon, exon)} \\
\mathbf{m}_{1:3} &= \alpha P(E_3 = g | X_3) \max_{x_2} (P(X_3 | x_2) \mathbf{m}_{1:2}) \\
&= \alpha \langle .25, .95, .1 \rangle \max_{x_2} \left(\begin{pmatrix} .75 & .25 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & .75 \end{pmatrix} \times \langle .1875, 0, 0 \rangle \right) \\
&= \alpha \langle .25, .95, .1 \rangle \max_{x_2} \begin{pmatrix} .141 & .047 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} = \langle .035, .045, .0 \rangle, \text{ (exon, exon, exon)} \\
\mathbf{m}_{1:4} &= \alpha P(E_4 = g | X_4) \max_{x_3} (P(X_4 | x_3) \mathbf{m}_{1:3}) \\
&= \alpha \langle .25, .95, .1 \rangle \max_{x_3} \left(\begin{pmatrix} .75 & .25 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & .75 \end{pmatrix} \times \langle .035, .045, 0 \rangle \right) \\
&= \alpha \langle .25, .95, .1 \rangle \max_{x_3} \begin{pmatrix} .026 & .009 & 0 \\ 0 & 0 & .045 \\ 0 & 0 & 0 \end{pmatrix} = \langle .0065, .0085, .0045 \rangle \text{ (exon, exon, splice)} \\
\mathbf{m}_{1:5} &= \alpha P(E_5 = a | X_5) \max_{x_4} (P(X_5 | x_4) \mathbf{m}_{1:4}) \\
&= \alpha \langle .25, .05, .4 \rangle \max_{x_4} \left(\begin{pmatrix} .75 & .25 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & .75 \end{pmatrix} \times \langle .0065, .0085, .0045 \rangle \right) \\
&= \alpha \langle .25, .05, .4 \rangle \max_{x_4} \begin{pmatrix} .0049 & .0016 & 0 \\ 0 & 0 & .0085 \\ 0 & 0 & .0034 \end{pmatrix} = \langle .0012, \approx 0, .0034 \rangle \text{ (exon, exon, splice)}
\end{aligned}$$

²Each row of the transition probability matrix is multiplied by the corresponding element of the message from $t - 1$. The maximum element of each column (corresponding to the max over x_{t-1} for each value of x_t) is then taken.

$$\begin{aligned}
\mathbf{m}_{1:6} &= \alpha P(E_6 = a | X_6) \max_{x_5} (P(X_6 | x_5) \mathbf{m}_{1:5}) \\
&= \alpha \langle .25, .05, .4 \rangle \max_{x_5} \left(\begin{pmatrix} .75 & .25 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & .75 \end{pmatrix} \times \langle .0012, 0, .0034 \rangle \right) \\
&= \alpha \langle .25, .05, .4 \rangle \max_{x_5} \begin{pmatrix} .0009 & .0003 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & .0026 \end{pmatrix} = \langle .0002, 0, .001 \rangle \text{ (exon, exon, intron)}
\end{aligned}$$

At this point, and until the final intron C, we will see intron as the likeliest prior state from which the transition to the next intron happened, so the most likely sequence (starting from $X_1 = \text{exon}$ that we knew) that generated our observations was:

⟨exon exon exon splice intron intron intron intron intron⟩.

And so the most likely splice site is the second g.

3 Markov Decision Processes

Suppose an agent inhabits a world with two states, s and $\neg s$. It can do exactly one of two actions, a and b . Action a does nothing and action b flips from one state to the other. The reward function is $R(s) = 3$, $R(\neg s) = 2$, and $\gamma = 0.5$. Complete the columns of the following table by performing policy iteration to find the optimal policy.

Answer:

	π^0	U^{π^0}	π^1	U^{π^1}	π^2
s	a	6	a	6	a
$\neg s$	a	4	b	5	b

4 Reinforcement Learning

1. You are a temporal difference agent in a completely observable environment. Your initial estimate for the utility $U(s)$ of states you haven't visited (initially all) is 0. Your future reward discount $\gamma = 1.0$. Your learning rate $\alpha = 0.1$. Here is the history of one trial taking you from state 1 to an enjoyable state 2, to a mildly painful terminal state 3 (It reads down the columns, left to right).

step	1	2	3
state s	1	2	3
reward R	-1	5	-4
action a	go2	go3	
new state s'	2	3	

What is your estimate of the utilities $U(s)$ for all states s at the end of step 1? At the end of step 2?

Answer: The TD equation for state values is $U(s) \leftarrow U(s) + \alpha(R(s) + \gamma U(s') - U(s))$. Thus after the first step, we update the value of state 1: $U(1) = 0 + .1(-1 + 0 - 0) = -.1$. After the second step, we update the value of state 2: $U(2) = 0 + .1(5 + 0 - 0) = .5$. At this point this value update does not propagate (yet) to state 1.

2. You are an Adaptive Dynamic Programming agent in the same environment. You make two trials, the first being the same as above in part 1. Here is the second:

step	1	2	3	4	5
state s	1	2	2	2	3
reward R	-1	5	5	5	-4
action a	go2	go2	go3	go3	
new state s'	2	2	2	3	

What is your estimate of the transition function (or table) $T(s, a, s')$ at the end of the first trial? At the end of the second trial?

Answer: The ADP agent estimates transition probabilities by adding up what happens and computing averages. After the first trial, we get $T(1, go2, 2) = 1$ and $T(2, go3, 3) = 1$. After the second trial, there is no change to $T(1, go2, 2)$. We add $T(2, go2, 2) = 1$ and revise the probabilities out of state 2: $T(2, go3, 3) = .5$ and $T(2, go3, 3) = .5$ after step 3 of trial 2, and revise them again to $T(2, go3, 2) = .33$ and $T(2, go3, 3) = .67$ after step 4 of trial 2.

3. How will you use the estimated values of $T(s, a, s')$?

Answer: The estimated transition probabilities and observed rewards are used to solve the corresponding MDP with value iteration or policy iteration.

5 More Reinforcement Learning

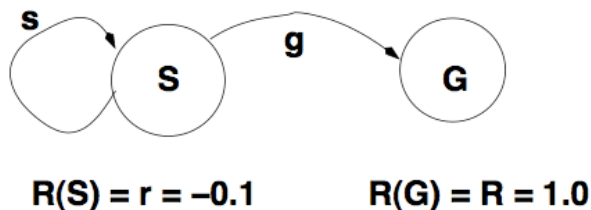


Figure 2: Simple MDP world.

The simple world in the figure has a start state S and a goal state G , with action s (for stay or start) that is meant to keep the agent in the start state, and g (for go or goal) meant to take agent to goal. The world is completely observable to the agent (it knows what state it is in and what reward it is getting). At goal, agent exits the game; there are no more moves until its next trial. Say there is a future-reward discount factor of γ . Reward of the start state is r and the reward of the goal is R , which in this world are -0.1 and $+1.0$ respectively. Both rewards and discounts start when the agent is born, so if the agent wakes up in S and takes action g successfully, its total reward is $0.1 + \gamma \times 1.0$. The world is uncertain. In fact the transition function $T(FromState, Action, ResultState)$ giving the probabilities of transitions given actions is: $T(S, s, S) = .9$, $T(S, s, G) = .1$, $T(S, g, S) = .2$, $T(S, g, G) = .8$. There are no $T(G, ., .)$ since state G is terminal. Call $T(S, g, G) = p$, the probability of successfully going, and then $T(S, g, S) = q = 1p$, the probability of an unsuccessful go action.

1. What is the agent's optimal policy?

Answer: The optimal policy is to always “go”, since “stay”ing incurs a cost.

2. What is the true utility of state S if $\gamma = 1$ (rewards are additive)? Give a numeric result.

Answer: The utility of a state is the expected sum of discounted rewards starting in that state and executing the optimal policy. If $\gamma = 1$ then it is the expected sum of (undiscounted) rewards. We compute the expectation over a reward by summing (over all states) the probabilities of transitioning to that state times the reward in that state. Note that in order to be in state S at time t , the agent needs to have transitioned t times from S to S (there is no transitioning back from G). And to be in G , the agent needs to have transitioned $t - 1$ times from S to S , then once from S to G . So:

$$\begin{aligned}
 U(S) &= E\left[\sum_{t=0}^{\infty} \text{Reward}(s_t)\right] = r + \sum_{t=1}^{\infty} (RP(s_t = G) + rP(s_t = S)) \\
 &= r + \sum_{t=1}^{\infty} q^{t-1}(pR + q^t r) \\
 &= -.1 + \sum_{t=1}^{\infty} .2^{t-1}(.8 \times 1 - .1 \times .2) = -.1 + .78 \times \frac{1}{1 - .2} = .875
 \end{aligned}$$

3. The agent does not know how the world works (transition probabilities) nor the utilities of any states. Suppose that by sheer luck, the agent is following the optimal policy. It makes four trials in the world;

in each trial it chooses the best action until it gets to G . The trials yield the following state sequences: 1) SSG, 2) SG, 3) SG, and 4) SSSG. What is the agent's resulting estimates of transition probabilities T ?

Answer: Count all transition occurrences $T(s, a, s')$, and normalize by the number of all transitions out of state s with action a . The agent is executing the optimal policy of always choosing the action "go". This will give $T(S, g, S) = 3/7$ and $T(S, g, G) = 4/7$. There are no transitions out of G , as G is a terminal state.

4. Now assume that the agent is a learner with a learning rate of $\alpha = .5$ and no discounting of rewards: $\gamma = 1$. After the first trial above, the agent computed state utilities by direct utility estimation. Then it became a temporal-difference learner, using TD utility updates after the second trial. What are the agent's state utility estimates after the first trial? After the second trial?

Answer: The first trial with utilities looked like this: $S_r S_r G_R$. After the first trial, the utility estimates using direct utility estimation (summing all observed rewards-to-go) are: $U(G) = 1$ and $U(S) = ((-.1 + 1) + (-.1 - .1 + 1))/2 = .85$. After the second trial, using TD learning:

$$\begin{aligned} U^\pi(S) &\leftarrow U^\pi(S) + \alpha(r + R + \gamma U^\pi(G) - U^\pi(S)) \\ &= .85 + .5(-.1 + 1 - .85) = .875 \end{aligned}$$

6 Bayes Net Inference

Consider the Bayes net shown below.

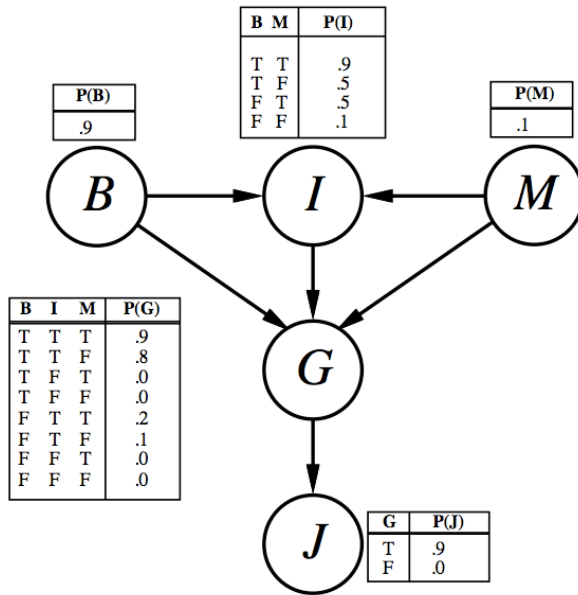


Figure 3: A simple Bayes net with boolean variables $B = BrokeElectionLaw$, $I = Indicted$, $M = PoliticallyMotivatedProsecutor$, $G = FoundGuilty$, and $J = Jailed$.

1. Which, if any, of the following are asserted by the network *structure* (ignoring the CPTs for now)?

- (i) $P(B, I, M) = P(B)P(I)P(M)$
- (ii) $P(J|G) = P(J|G, I)$
- (iii) $P(M|G, B, I) = P(M|G, B, I, J)$

Answer: Both (ii) and (iii) are true. For (iii), consider the Markov blanket of M .

2. Calculate the value of $P(b, i, \neg m, g, j)$

Answer: $P(b, i, \neg m, g, j) = P(b)P(\neg m)P(i|b, \neg m)P(g|b, i, \neg m)P(j|g) = .9 \times .9 \times .5 \times .8 \times .9 = .2916$

3. Calculate the probability that someone goes to jail given that they broke the law, have been indicted and face a politically motivated prosecutor.

Answer: Since B , I , and M are fixed true in the evidence, we can treat G as having a prior of .9 and just look at the submodel with G and J :

$$\begin{aligned}
P(J|b, i, m) &= \alpha \sum_g P(J, g) = \alpha[P(J, g) + P(J, \neg g)] \\
&= \alpha[\langle P(j, g), P(\neg j, g) \rangle + \langle P(j, \neg g), P(\neg j, \neg g) \rangle] \\
&= \alpha[\langle .81, .09 \rangle + \langle 0, .1 \rangle] = \langle .81, .19 \rangle
\end{aligned}$$

That is, the probability of going to jail given the situation described is 0.81.

4. A *context-specific* independence has the following form: X is conditionally independent of Y given Z in context $C = c$ if $P(X|Y, Z, C = c) = P(X|Z, C = c)$. In addition to the usual conditional independences given by the graph structure, what context-specific independences exist in the Bayes net shown in the figure above?

Answer: Intuitively, a person cannot be found guilty if not indicted, regardless of whether they broke the law and regardless of the prosecutor. This is what the CPT for G says; so G is context-specifically independent of B and M given $I = false$.

5. Suppose we want to add the variable $P = PresidentialPardon$ to the networks; draw the new network and briefly explain any links you add.

Answer: A pardon is unnecessary if the person is not indicted or not found guilty. So I and G are parents of P . One could also add B and M as parents of P , since a pardon is more likely if the person was actually innocent (did not break the law) or if the prosecutor is politically motivated. The pardon is a get-out-of-jail-free card, so P is a parent of J .

7 Perceptrons

1. The following table shows a data set and the number of times each point is misclassified during a run of the perceptron algorithm, starting with zero weights. What is the equation of the separating line found by the algorithm, as a function of x_1 , x_2 , and x_3 ?

x_1	x_2	x_3	d	times misclassified
2	3	1	+1	12
2	4	0	+1	0
3	1	1	-1	3
1	1	0	-1	6
1	2	1	-1	11

Assume that the learning rate is 1 and the initial weights are all zero.

Note: I changed the fourth column of the table from y to d for consistency: y is the perceptron output, d is the desired output label given in the training set.

Answer: A perceptron “embodies” the linear separator (line, plane, hyperplane) that it uses to classify input examples. So the equation of the separating line (which is actually a plane in this case, since the inputs are 3-dimensional) is where the weighted sum of inputs (and bias) is equal to 0: $-w_0 + w_1x_1 + w_2x_2 + w_3x_3 = 0$. To find that, we need to find the weights that the perceptron learns from the training data set we were given.

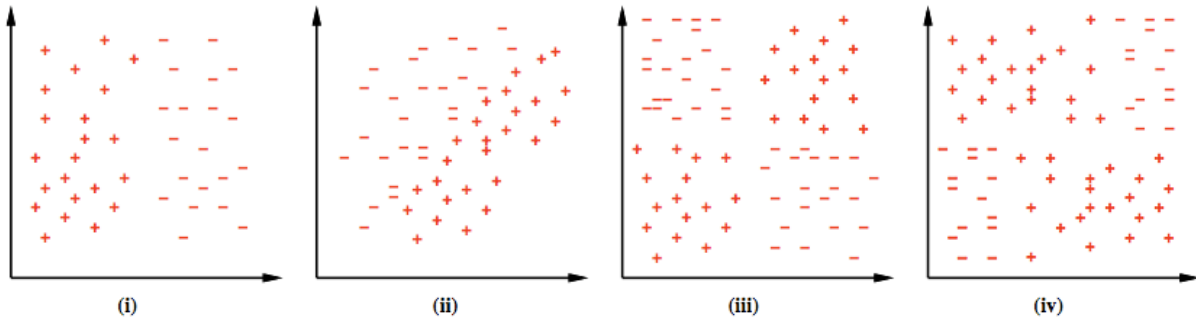
$$\begin{aligned} \forall j: w_j &= w_j^{init} + \alpha \sum_{i=1}^m (d^i - y^i) x_j^i \\ &= \sum_{i=1}^m (d^i - y^i) x_j^i \end{aligned}$$

Because initial weights are 0 and the learning rate is 1, we just need to add up all the updates from each of the misclassified example in the training set.

$$\begin{aligned} w_0 &= \sum_{i=1}^m (d^i - y^i)(-1) \\ &= -1(2 \times 12 + (-2) \times (3 + 6 + 11)) = -16 \\ w_1 &= \sum_{i=1}^m (d^i - y^i)x_1 \\ &= 2 \times 2 \times 12 + (-2) \times 3 \times 3 + (-2) \times 1 \times 6 + (-2) \times 1 \times 11 = -4 \\ w_2 &= \sum_{i=1}^m (d^i - y^i)x_2 \\ &= 2 \times 3 \times 12 + (-2) \times 1 \times 3 + (-2) \times 1 \times 6 + (-2) \times 2 \times 11 = 10 \\ w_3 &= \sum_{i=1}^m (d^i - y^i)x_3 \\ &= 2 \times 1 \times 12 + (-2) \times 1 \times 3 + (-2) \times 0 \times 6 + (-2) \times 1 \times 11 = -4 \end{aligned}$$

So the equation of the separating line is: $-16 - 4x_1 + 10x_2 - 4x_3 = 0$ (can also divide through by 2).

2. Which of the four data sets below are correctly classifiable with a perceptron? The inputs have two continuous features. The two classes are labelled by + and -.



Answer: (i) and (ii) are correctly classifiable by a perceptron, because they are linearly separable. The other two are not.

8 Overfitting and Complexity

1. How does a neural net overfit? How can you reduce overfitting in a neural net?

Answer: By having too many units, and therefore too many weights, thus enabling it to fit every nuance of the training set. Also, by training too long so as to fit the training data better.

You can reduce overfitting by using cross-validation to choose a not too complex network. And by using a validation (test) set to decide when to stop training.

2. How does a Bayesian classifier overfit? How can you reduce overfitting?

Answer: By using the wrong prior, or a uniform prior over a large hypothesis space with little data. Reduce overfitting by using good prior information (no free lunch theorem).

3. Which of the following approaches do perceptrons use to control complexity while minimizing error? Which do feed-forward neural networks (with no weight decay or early stopping) use?

- A: Use a fixed-complexity hypothesis class
- B: Include a complexity penalty in the measure of error
- C: Nothing

Answer: A. The perceptron uses a fixed hypothesis class of linear separators. Feed-forward neural networks also use a fixed hypothesis class, which is determined by the wiring diagram of the network.

9 Naive Bayes

Consider a Naive Bayes problem with three boolean features, $x_1 \dots x_3$. Imagine that we have seen a total of 12 training examples, 6 positive (with $y = 1$) and 6 negative (with $y = 0$). Here is a table with some of the counts:

	$y = 1$	$y = 0$
$x_1 = 1$	5	5
$x_2 = 1$	0	0
$x_3 = 1$	2	4

1. Supply the following estimated probabilities:

Answer:

$$P(x_1 = 1|y = 0) = \frac{5}{6}$$

$$P(x_2 = 1|y = 1) = \frac{0}{6} = 0$$

$$P(x_3 = 0|y = 0) = 1 - \frac{2}{6} = \frac{2}{3}$$

2. Is there anything problematic with any of the above estimates? Be specific.

Answer: Because in 12 examples, we have not seen any with a positive x_2 , those probability estimates are set to 0. This is problematic because a small data set is likely to not contain all types of features, even if they occur with positive probabilities.

3. Which feature plays the largest role in deciding the class of a new instance? Why?

Answer: x_3 , because it has the biggest difference in the likelihood of being true for the two different classes. The other two features carry no information about the class.

10 Bayesian learning

Suppose that you want to build a program that detects whether an incoming e-mail message is spam or not. You decide to attack this using machine learning. So, you collect a large number of training messages and label them as spam or not-spam. You further decide that you will use the presence of individual words in the body of the message as features. That is, you collect every distinct word found in the training set and assign to each one an index, from 1 to N . You count the number of times each word occurred in the training set messages that were spam and not-spam. Then, given a message, you construct a feature vector with N entries and write in each entry a 1 if the word appears in the message, and 0 if it does not.

1. What does a naive Bayes model for this problem look like?

Answer: A naive Bayes model has one cause variable $S = Spam$ and N effect variables W_1 through W_N each for the presence or absence of a particular word in a message. The only arrows go from S to each of the W_i , as word occurrence is assumed independent.

2. Assuming a naive Bayes model, how would you compute its parameters? Give precise formulas.

Answer: The model has $1 + 2N$ parameters: one for the prior $\theta = P(S = spam)$ and two each for each word W_i : $\theta_{i1} = P(W_i = 1|spam)$ and $\theta_{i2} = P(W_i = 1|notspam)$. To estimate these parameters from the training data set:

$$\theta = \frac{\#spam}{\#messages}$$

$$\theta_{i1} = \frac{\#W_i \text{ in spam}}{\#spam}$$

$$\theta_{i2} = \frac{\#W_i \text{ in notspam}}{\#notspam}$$

3. What assumption did you make in your calculations?

Answer: A uniform prior over the hypothesis space. This is a maximum-likelihood estimation.

4. What would be a less naive model?

Answer: It would be less naive to not assume independent word occurrence in a message. For example, you could draw arrows between words that have the same root, like “win” and “winner”.