Homework 2: Discrete Variables & Independence

UC Irvine CS177: Applications of Probability in Computer Science

Due on October 17, 2019 at 11:59pm

Question 1: (20 points)

A recent study has shown that 10% of computer science concentrators suffer from symptoms of Carpal Tunnel syndrome. Researchers have developed a diagnostic test for this condition and in a recent trial, the test gave a positive result in 96% of the patients who were known to truly have Carpal Tunnel syndrome.

a) *For an initial version of the diagnostic test, the false positive rate (the percentage of people without Carpal Tunnel who incorrectly test positive) was 44%. If a student received a positive result from this test, what would be the posterior probability that they truly have Carpal Tunnel syndrome?*

b) *The researchers believe their test will be practically useful if a positive result implies the student has Carpal Tunnel with at least 75% probability. What is the largest possible false positive rate that would achieve this accuracy target?*

Question 2: (20 points)

In a casino game, gamblers are allowed to roll n fair, 6-sided dice. If a 6 shows up on any of them, the gambler gets nothing. If no 6's appear, the gambler is paid the sum of the values on the dice in dollars. The gambler is free to choose n, the number of dice rolls.

a) *Let R_n be the event that in n dice rolls, all rolls are 5 or less (no 6's appear). Compute $P(R_n)$ as a function of n.*

b) *Let S_n be the payoff (the number of dollars earned) after n dice rolls. Compute $E[S_n \mid R_n]$, the expected payoff given that no 6's appear, as a function of n.*

c) *Using the results from parts (a) and (b), derive a formula for the gambler's expected payoff $E[S_n]$. Plot this payoff for values of n from 1 to 20. What is the smallest n that maximizes the expected payoff?*

Question 3: (20 points)

We consider a probabilistic model for a fault diagnosis problem. The class variable C represents the health of a disk drive: $C = 0$ means it is operating normally, and $C = 1$ means it is in a failed state. When the drive is running it continuously monitors itself using a temperature and shock sensor, and records two binary features, X and Y. $X = 1$ if the drive has been subject to shock (e.g., dropped), and $X = 0$ otherwise. $Y = 1$ if the drive temperature has ever been above 70°C, and $Y = 0$ otherwise. The following table defines the joint probability mass function of these three random variables:

x	y	c	$p_{XYC}(x, y, c)$
0	0	0	0.1
0	1	0	0.2
1	0	0	0.2
1	1	0	0.1
0	0	1	0.0
0	1	1	0.1
1	0	1	0.05
1	1	1	0.25

When computing probabilities below, provide the numerical value of your answer, as well as equations showing how you calculated that answer.

a) What is the probability $P(C = 1)$*?*

b) What is the probability $P(C = 0 \mid X = 1, Y = 0)$*?*

c) What is the probability $P(X = 0, Y = 0)$*?*

d) What is the probability $P(C = 0 \mid X = 0)$*?*

e) Are X *and* Y *independent? Justify your answer.*

f) Are X *and* Y *conditionally independent given* C*? Justify your answer.*

Question 4: (40 points)

As a faculty member at UC Irvine, Erik receives a *lot* of email, and he is in desperate need of a method for separating important messages from spam. Now that you've learned some discrete probability, could you help him out?

Our dataset was released during the "Enron" corruption investigation, and contains emails with labels of *spam* or *ham*. You'll use a "naive Bayes" classifier to identify spam emails. The vocabulary `vocab` consists of W words, where each character string has been mapped to a distinct integer index. The training data matrix `trainFeat` is a $D \times W$ matrix, with each row represents an email and each column indicates whether that word appears in that email at least once. The ground truth labels are stored in `trainLabels`, with "1" indicating spam and "0" ham. Test data, to be used for evaluating performance but *not* estimating probabilities, is stored in `testFeat`, `testLabels`.

To define a probabilistic model of this data, we let $Y_i = S$ if email i is spam, and $Y_i = H$ if email i is ham (not spam). We assume that the two classes are equally likely *a priori*:

$$P(Y_i = S) = P(Y_i = H) = 0.5. \tag{1}$$

To encode the data that will be used for classification, we let $X_{ij} = 1$ if email i contains an instance of word j, and $X_{ij} = 0$ if email i does *not* contain word j. The set of all available data about email i is then $X_i = \{X_{ij} \mid j = 1, \ldots, W\}$.

To implement a simple Bayesian classifier, we will compute the posterior distribution $P(Y_i \mid X_i)$ of the class label given the observed features. If $P(Y_i = S \mid X_i) > P(Y_i = H \mid X_i)$, we classify email i as spam. Otherwise, we classify it as ham.

a) *The Bayesian classifier described above is equivalent to a classifier that assigns label spam if $P(X_i \mid Y_i = S) > P(X_i \mid Y_i = H)$, and label ham otherwise. Using Bayes' rule and Equation* (1)*, give a simple argument for why this is true.*

To further simplify the modeling problem, a *naive Bayes classifier* assumes that given the class label, the observed word features are conditionally independent. From the definition of independence, this implies that

$$P(X_i \mid Y_i = S) = \prod_{j=1}^{W} P(X_{ij} \mid Y_i = S), \qquad P(X_i \mid Y_i = H) = \prod_{j=1}^{W} P(X_{ij} \mid Y_i = H). \tag{2}$$

b) *A simple way to estimate the probabilities above is by counting how many times each event occurs in the training data. Let N_s be the total number of spam emails, N_{sj} the number of spam emails in which word j occurs, N_h the total number of ham emails, and N_{hj} the number of ham emails in which word j occurs. We then set*

$$P(X_{ij} = 1 \mid Y_i = S) = \frac{N_{sj}}{N_s}, \quad P(X_{ij} = 0 \mid Y_i = S) = 1 - P(X_{ij} = 1 \mid Y_i = S) = \frac{N_s - N_{sj}}{N_s},$$
$$P(X_{ij} = 1 \mid Y_i = H) = \frac{N_{hj}}{N_h}, \quad P(X_{ij} = 0 \mid Y_i = H) = 1 - P(X_{ij} = 1 \mid Y_i = H) = \frac{N_h - N_{hj}}{N_h}.$$

Write Python code to compute these probabilities using data in `trainFeat,trainLabels`*.*

c) *Consider a simplified dataset that only contains the presence or absence of a single word, j="money". Compute and report the numerical values of the conditional probabilities* $P(X_{ij} = 1 \mid Y_i = S)$, $P(X_{ij} = 1 \mid Y_i = H)$. *What is the test accuracy of a Bayesian classifier based on this single word?*

d) *Repeat part (c) for a different single word, j="thanks". Provide an intuitive explanation for any differences in classification performance.*

e) *Repeat part (c) for a different single word, j="possibilities". Provide an intuitive explanation for any differences in classification performance.*

f) *Consider a slightly larger dataset which contains the presence or absence of the three words ("money,thanks,possibilities") from parts (c-e). Using the naive Bayes assumption from Equation* (2)*, determine the test accuracy of a classifier based on these three words.*

When the number of words W is large the probabilities of Equation (2) become very small, and can underflow to zero when using finite-precision arithmetic on a computer. To avoid this, we will instead work in the log-domain, and pick the class whose log-probability is largest. Because a log-of-products is equal to a sum-of-logs, we have:

$$\ln P(X_i \mid Y_i = S) = \ln \prod_{j=1}^{W} P(X_{ij} \mid Y_i = S) = \sum_{j=1}^{W} \ln P(X_{ij} \mid Y_i = S), \tag{3}$$

with a similar identity for $\ln P(X_i \mid Y_i = H)$.

g) *Using the identity in Eq.* (3)*, modify your classification code to compute the log-probability of the spam and ham classes in a numerically robust fashion. Determine the test accuracy of a classifier based on all* W *words in the full dataset.* HINT: This classifier should take seconds (not minutes) to train and test, and be much more accurate than part (f).