STAT 339: HOMEWORK 6 (BAYES NETS AND BAYESIAN CLASSIFICATION)

DUE VIA GITHUB FRIDAY 1/7/21

Instructions. Create a directory called `hw6` in your `stat339` GitHub repo. Your main writeup should be called `hw6.pdf`.

You may also use any typesetting software to prepare your writeup, but the final document should be a PDF. LaTeXis highly encouraged.

I will access your work by cloning your repository; make sure that any file path information is written relative to your repo – don't use absolute paths on your machine, or the code won't run for me!

Date: Last Revised: December 19, 2021.

1. Bayes Nets

0. Consider the Bayes net depicted in Fig. 1, which comes from the BRML book. Each variable is binary.

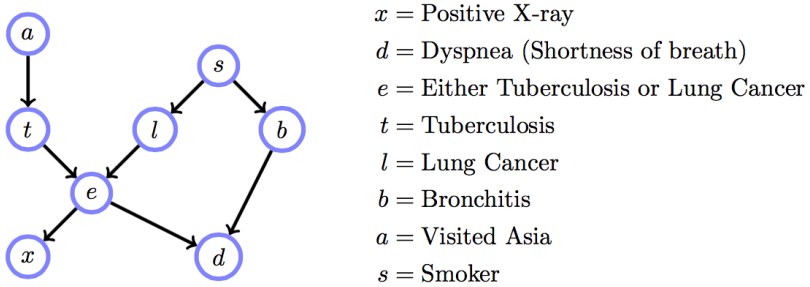

Figure 3.15: Belief network structure for the Chest Clinic example.

FIGURE 1. Bayes Net for diagnosis of lung disease at a chest clinic

(a) Write down the factorization of the joint distribution that is implied by the graph.

(b) According to the model, can you predict **whether someone has visited Asia** based on **whether or not they are a smoker**? That is, **are** s **and** a **independent**?

(c) Does knowing that someone **is a smoker** help you predict whether they **visited Asia** if you also have a **chest x-ray**? That is, are s and a **conditionally independent given** x? Explain the intuition behind these two results.

2. Naive Bayes With Categorical Features

1. **Spam Filtering** (Adapted from BRML 10.5) This problems is about a hypothetical classifier to label emails as either "spam" or "not spam". The questions do not involve actually implementing the classifier, just examining and reflecting on its mathematical/statistical properties.

   Each email is represented by a vector of **binary** features:

   $$\mathbf{x}_n = (x_{n1}, ..., x_{nD})$$

   where each $x_{nd} \in \{0, 1\}$. Each entry of the vector indicates *whether* a particular symbol or word (out of D symbols/words in the vocabulary) appears in the email. The symbols/words are things like

   $$money, cash, !!!, viagra, \ldots, etc.$$

   so that, for example, $x_{n2} = 1$ if the word 'cash' appears in email n (**Note:** this is a **different** way of representing the contents of a document than the Federalist papers example from class, though the basic classification goal is essentially the same)

   The training dataset consists of a set of vectors along with the class label t_n for each email, where $t_n = 1$ indicates that email n is spam, and $t_n = 0$ indicates that it is not spam. Therefore, the training set consists of a set of pairs $\{(\mathbf{x}_n, t_n)\}, n = 1, \ldots, N$.

   The naive Bayes model for the **joint probability** of the **category** (t_n) and **contents** (abstracted as $\mathbf{x}_n$) of email n is

   $$p(t_n, \mathbf{x}_n \mid \boldsymbol{\theta}, \pi) = p(t_n \mid \pi) \prod_{d=1}^{D} p(x_{nd} \mid t_n, \boldsymbol{\theta})$$

   Explicitly, the parameters are $(\pi, \theta_{01}, \ldots, \theta_{0D}, \theta_{11}, \ldots, \theta_{1D})$, where

   $$\begin{aligned} \pi &:= p(t_n = 1 \mid \pi), && \text{for all } n \\ \theta_{1d} &:= p(x_{nd} = 1 \mid t_n = 1, \boldsymbol{\theta}) && \text{for all } n \\ \theta_{0d} &:= p(x_{nd} = 1 \mid t_n = 0, \boldsymbol{\theta}) && \text{for all } n \end{aligned}$$

   That is to say, each $t_n \mid \pi \sim \mathsf{Bernoulli}(\pi)$, and each $x_{nd} \mid t_n = c, \boldsymbol{\theta} \sim \mathsf{Bernoulli}(\theta_{cd})$: The same parameters are assumed to apply for every email of the same type (spam or not spam), which is why n does not appear in their definitions.

(a) Derive expressions for the **maximum likelihood estimates** of $\boldsymbol{\theta}$ and π, in terms of of the training data. Assume that, the collection of labels t_ns are **conditionally independent given** π, and that the $\mathbf{x}_n$ are conditionally independent of each other given the t_n and $\boldsymbol{\theta}$. That is, assume

$$p(t_1, ..., t_N, \mathbf{x}_1, ..., \mathbf{x}_N \mid \pi, \boldsymbol{\theta}) = \prod_{n=1}^{N} p(t_n \mid \pi) \left(\prod_{d=1}^{D} p(x_{nd} \mid t_n, \boldsymbol{\theta}) \right)$$

(b) Given a **trained model** (i.e., given MLEs for the $\hat{\pi}_{\text{MLE}}$ and $\hat{\boldsymbol{\theta}}_{\text{MLE}}$ parameters), **give an expression for the posterior probability** that a new email is spam, that is, for $p(t_{\text{new}} = 1 \mid \mathbf{x}_{\text{new}}, \hat{\boldsymbol{\theta}}_{\text{MLE}}, \hat{\pi}_{\text{MLE}})$ where t_{new} and $\mathbf{x}_{\text{new}}$ are the category and feature vector, respectively, for a new email. The expression should be explicitly stated in terms of π_{MLE}, the entries of $\hat{\boldsymbol{\theta}}_{\text{MLE}}$, and the binary entries of $\mathbf{x}_{\text{new}}$ only, such that if you had numbers for each of these things, you could plug them in to calculate a numerical value for the posterior probability.

(c) If the word "viagra" **never appears in the spam training data**, discuss **what effect this will have** on the classification for a **new email** that contains the word "viagra", assuming we are using the MLE parameter estimates. Explain **how you might counter this effect**.

(d) What effect will **misspelled words** (such as "v1agra") have on the spam filter? **How could a spammer try to fool** a naive Bayes spam filter **if they know that the spam filter is a naive Bayes classifier**?

2. **Naive Bayes Classification as Regression** Show that, when using the naive Bayes classifier above, for fixed θ and π, the **log odds** that an email is spam, defined as

$$\text{logodds}(t_n = 1 \mid \mathbf{x}_n, \pi, \boldsymbol{\theta}) := \log \left(\frac{p(t_n = 1 \mid \mathbf{x}_n, \pi, \boldsymbol{\theta})}{p(t_n = 0 \mid \mathbf{x}_n, \pi, \boldsymbol{\theta})} \right)$$

can be written as

$$\text{logodds}(t_n = 1 \mid \mathbf{x}_n, \pi, \boldsymbol{\theta}) = w_0(\boldsymbol{\theta}, \pi) + \sum_{d=1}^{D} w_d(\boldsymbol{\theta}, \pi) x_{nd}$$

for some suitably chosen **weight functions** w_d, $d = 0, \ldots, D$, of the parameters, π and $\boldsymbol{\theta}$ (which do **not** depend on the data, provided we have chosen values for π and $\boldsymbol{\theta}$). That is, the log odds that the email is spam is a **linear**

function of the entries in $\mathbf{x}_n$. **Find explicit expressions for these weight functions** $w_0, w_1, \ldots, w_D$s in terms of π and the entries in $\boldsymbol{\theta}$ only.

3. **Naive Bayes for Cancer Screening** The data for this problem consists of several diagnostic variables from tumors from each of 699 breast cancer patients (modified from a dataset in the University of California Irvine Machine Learning Repository[1]).

   - The class variable, t, is binary: Is the tumor malignant?
   - The nine diagnostic variables (which make up the 699×9 feature matrix $\mathbf{X}$) are measurements of things like **mean cell size**, **variability of cell sizes**, various **shape measures**, etc. Each diagnostic variable has been coded on an **integer scale ranging from 1 to 10**.

   I have randomly divided the full dataset into **training** and **test** sets: `cancer_train.csv` and `cancer_test.csv`, containing 2/3 and 1/3 of the cases, respectively. In row n of the `.csv` file:

   - The **first entry** is an ID code (don't use this for classification)
   - The second is the **target**, t_n, the binary `Malignant` label (0 or 1)
   - The remaining columns are the **diagnostic features**, where each x_{nd} has a value in the set $\{1, 2, \ldots, 10\}$, for $n = 1$ to 699 and $d = 1$ to 9, with the exception of missing values (see below).

   Some of the cases have missing values for one of the features, `BareNuclei`. These missing values are denoted by -1 in the data. **Be sure to handle these as missing, not as a data value**. Note also that for several features, **not all of the values 1-10 might appear in both tumor types**, but they could in principle.

   Your mission (should you choose to accept it) is to **design a naive Bayes classifier that reports, for a novel case, a probability that it is malignant**. In order to do this, **you will need to make some subjective design decisions** about how to represent the data-generating process.

   You may choose to use the feature values as they are, or to bin them (since they consist of ordered values). If you choose to bin, you might select bins that have equal numbers of feature values, or bins that have approximately

[1] `http://archive.ics.uci.edu/ml/`

equal numbers of cases aggregated over classes, or use some other scheme; up to you.

(a) Implement a training function, `train_naive_bayes()`, that takes in a set of **training data** and returns a **classifier function**. The classifier function should take an $N \times D$ matrix $\mathbf{X}_{\text{new}}$ as input and return an $N \times 2$ **array of probabilities**, where the entry in row n and column c is the posterior probability that case n has class c.

It is up to you if you want your function to take a `.csv` file consisting of training data directly, or preprocess the data first and pass in $\mathbf{t}$ and $\mathbf{X}$ as separate arguments — probably the latter will make your code more generalizeable)

Your training function should have two modes, which can be selected via an argument. In the first mode it should **find the maximum likelihood estimates** of the prevalence parameter π ($\pi_c := p(t = c \mid \pi)$) and the class conditional distribution parameters $\boldsymbol{\theta}$, and should return a classifier that uses these to classify the new data.

Note that because the features (columns of the input matrix) are **different kinds of things**, we probably want to have a separate probability vector $\boldsymbol{\theta}_{cd}$ for each feature for each class. That is, we should let $\theta_{cdk} := p(x_{nd} = k \mid t_n = c, \boldsymbol{\theta}_{cd})$, where k indexes the possible values of feature d (however you decided to bin them, if you did, or just 1 through 10 if you didn't) and $\sum_{k=1}^{K} \theta_{cdk} = 1$ for each c, d.

In the second mode, the user should be able to specify **Dirichlet priors** for $\boldsymbol{\pi}$ and for each $\boldsymbol{\theta}_{cd}$, and the resulting classifier should return the array of **posterior predictive probabilities** that each tumor in $\mathbf{X}_{\text{new}}$ belongs to each category.

Recall that this is defined as

$$p(t_{\text{new}} = c, \mid \mathbf{x}_{\text{new}}, \mathbf{t}_{\text{train}}, \mathbf{X}_{\text{train}}) = \int \int p(t_{\text{new}} = c \mid \mathbf{x}_{\text{new}}, \boldsymbol{\theta}, \boldsymbol{\pi}) p(\boldsymbol{\pi}, \boldsymbol{\theta} \mid \mathbf{t}_{\text{train}}, \mathbf{X}_{\text{train}}) d\boldsymbol{\pi} d\boldsymbol{\theta}$$

where the first factor inside the integral is the **posterior probability** that the new tumor belongs to class c for specific parameter settings $\boldsymbol{\pi}$ and $\boldsymbol{\theta}$, and the second factor represents the posterior density of that combination of $\boldsymbol{\pi}$ and $\boldsymbol{\theta}$. However, since we are using a **conjugate prior**, the result of this integral has a very simple form, which we derived in class (and so you do not actually need to work with this integral!).

The specification of the parameters of the Dirichlet priors on $\boldsymbol{\pi}$ and each $\boldsymbol{\theta}_{cd}$ could in principle involve separate parameter vectors for each, but to simplify things assume that all of the priors on the $\boldsymbol{\theta}_{cd}$s are **symmetric Dirichlet** distributions, that is that

$$\boldsymbol{\theta}_{cd} \sim \mathsf{Dir}(\alpha/K, \alpha/K, \ldots, \alpha/K)$$

governed by a single scalar parameter α. However, the prior on $\boldsymbol{\pi}$ should be allowed to be an arbitrary $\mathsf{Dir}(\gamma_1, \ldots, \gamma_C)$ distribution over C probabilities ($C = 2$ for this data), since we likely do not expect malignant and non-malignant tumors to be equally common.

(b) **Explain the shortcomings of maximum likelihood estimation** when it comes to the possibility of seeing a feature take the value k in the test set that did not appear in any of the cases in the training set.

(c) Discuss **why a naive Bayes classifier trivially handles missing features**, whereas a KNN classifier would have problems

(d) For the Bayesian method, **use cross-validation to find the best choice of the prior parameter** α on the Dirichlet priors on the $\boldsymbol{\theta}_{cd}$s. Here, "best" is defined in terms of the mean cross-validation error. You do not need to worry about finding the best γs empirically — you can treat these as fixed.