STAT 339: HOMEWORK 4 (BAYESIAN CLASSIFICATION AND REGRESSION)

UPDATED: DUE VIA GITHUB MONDAY 4/27)

Instructions. Create a directory called `hw4` in your `stat339` GitHub repo. Your main writeup should be called `hw4.pdf`, and any source code should either be in that directory, a subdirectory within it, or a "library" directory at the top level (in the case of files defining functions used in multiple assignments).

I suggest placing the definitions of any "helper" functions in a separate file which you load (in Python, `import`) from your main file.

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!

You may use any language you like to do this assignment — the tasks are stated in a language-neutral way — but Python is recommended.

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

All data files referred to in the problems below can be found at

`http://colindawson.net/data/<filename>.csv`.

Date: Last Revised: April 12, 2020.

1. Naive Bayes With Categorical Features

1. **Spam Filtering** (Adapted from BRML 10.5) This question is about a classifier to label emails as either "spam" or "not spam".

   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.

   The training dataset consists of a set of vectors along with the class label t, where $t = 1$ indicates the email is spam, and $t = 0$ 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 and contents 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

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

   (That is to say, each $t_n \mid \pi \sim \mathsf{Bern}(\pi)$, and each $x_{nd} \mid t_n, \boldsymbol{\theta} \sim \mathsf{Bern}(\theta_{t_n,d})$: 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 data is independent and identically distributed; that is that

   $$p(t_1, ..., t_N, \mathbf{x}_1, ..., \mathbf{x}_N \mid \pi, \boldsymbol{\theta}) = \prod_{n=1}^{N} p(t_n, \mathbf{x}_n \mid \pi, \boldsymbol{\theta})$$

(b) Given a trained model (i.e., given MLEs for the $\hat{\pi}_{\text{MLE}}$ and $\hat{\boldsymbol{\theta}}_{\text{MLE}}$ parameters), explain how to find $p(t_{new} \mid \mathbf{x}_{new}, \hat{\boldsymbol{\theta}}_{\text{MLE}}, \hat{\pi}_{\text{MLE}})$.

(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". 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

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

is a linear function of the individual x_{nd}; that is, that we can write

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

where the w_ds, $d = 0, \ldots, D$, are functions of the parameters, π and $\boldsymbol{\theta}$. Find expressions for these ws.

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 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.

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

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:

- The first entry is an ID code
- The second is t_n, the binary `Malignant` label (0 or 1)
- The remaining columns are the diagnostic features, $x_{nd} \in \{1, 2, \ldots, 10\}$.

Note that 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 appear in both tumor types.

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 equal numbers of cases aggregated over classes.

(a) Implement a training function that takes in the data and returns maximum likelihood estimates of the **prevalence** ($\pi := P(t = \text{“}malignant\text{”} \mid \pi)$) and of the parameters $\boldsymbol{\theta}_{td} = (\theta_{td1}, \ldots, \theta_{tdK})$ of the (Categorical) class conditional distributions, for each diagnostic feature: $p(x_{nd} = k \mid t, \boldsymbol{\theta}_{td}) = \theta_{ndk}$, $k = 1, \ldots, K$.

(b) Implement a function that takes in the data as well as prior parameters for for a $\mathsf{Beta}(a, b)$ prior on π and identical symmetric Dirichlet priors on each $\boldsymbol{\theta}_{td}$ (that is, assume that the prior on each $\boldsymbol{\theta}_{td}$ is

$$\boldsymbol{\theta}_{td} \sim \text{Dirichlet}(\alpha_{td1}, \alpha_{td2}, \ldots, \alpha_{tdK})$$

where $\alpha_{tdk} \equiv \alpha$ for all t, d, k) and returns posterior parameters.

The Beta prior on π has parameters a and b, and the Dirichlet priors on the $\boldsymbol{\theta}_{td}$ share a single parameter α. So all together you will specify scalars a and b and α representing the prior parameters, and return scalars $a^{(post)}$ and $b^{(post)}$, and a $2 \times D \times K$ array, $\boldsymbol{\alpha}$ with entries $\alpha_{t,d,k}^{(post)}$ representing the posterior parameters (these will not be identical even though the prior values were).

(c) Implement a classifier function that takes in a new feature vector $\mathbf{x}_{\text{new}}$ as well as point estimates for π, $\boldsymbol{\theta}_0$ and $\boldsymbol{\theta}_1$ (as in the MLE formulation), and returns the probability that the tumor is malignant.

(d) Implement a variant of your classifier that takes in a feature vector $\mathbf{x}_{\text{new}}$, as well as values of $a^{(post)}$, $b^{(post)}$, and the array $\boldsymbol{\alpha}$ of $\alpha_{t,d,k}^{(post)}$ values, and returns the **posterior predictive probability** that the tumor is malignant. Recall that this is

$$p(t_{\text{new}} = 1 \mid \mathbf{x}_{\text{new}}) = \int\int p(t_{\text{new}} = 1 \mid \pi) p(\mathbf{x} \mid t_{\text{new}} = 1, \boldsymbol{\theta}) p(\pi \mid a^{(post)}, b^{(post)}) p(\boldsymbol{\theta} \mid \boldsymbol{\alpha}) d\pi d\boldsymbol{\theta}$$

(e) In order to guide treatment or further diagnostic options, physicians will likely want a binary decision from your classifier: "is the test positive?". Note that, unlike in the digits problem on HW1, the cost associated with a false positive is likely different than the cost associated with a false negative, so returning the label with the highest posterior probability may not be the best choice. Instead, **write a function that takes in the posterior predictive probability that a tumor is malignant and allows a user to specify a loss function**, $L(\hat{t}, t)$ by specifying the cost associated with a false positive ($L(1, 0)$) and the cost associated with a false negative ($L(0, 1)$) – assume the cost of a correct classification is 0 – **and returns the label with the lowest expected cost**; i.e. return

$$t^* = \arg\min_{\hat{t}} \mathbb{E}\left[L(\hat{t}, t) \mid \mathbf{x}\right] = \arg\min_{\hat{t}} \sum_{t} L(\hat{t}, t) p(t \mid \mathbf{x})$$

(f) **Explain the shortcomings of maximum likelihood estimation when it comes to the zero counts**.

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

(h) For the Bayesian method, **use cross-validation to find the best choice of the prior parameter α, where "best" is defined as minimizing the actual realized cost**

$$C(\alpha) = \sum_{n \in \text{Validation}} L(t_n, t_n^*)$$

where t^* and L are defined in part 3e. We can think of this value as a "smoothing" term to deal with sparse data, playing an analogous role to the ridge parameter λ in ridge regression.

4. **Bayesian Polynomial Regression** Consider the generative linear model:

$$\mathbf{t} = \mathbf{Q}^*\mathbf{w} + \boldsymbol{\varepsilon}$$

where $\mathbf{Q}^* = \mathbf{Q}\sqrt{N-1} = \mathbf{X}\mathbf{R}^{-1}\sqrt{N-1}$, for $\mathbf{Q}$ and $\mathbf{R}$ obtained by QR decomposition of the $N \times (D+1)$ feature matrix $\mathbf{X}$ (recall that $\mathbf{Q}$ satisfies $\mathbf{Q}^\mathsf{T}\mathbf{Q} = \mathbf{I}$, and therefore $(\mathbf{Q}^*)^\mathsf{T}\mathbf{Q}^* = (N-1)\mathbf{I}$), and

$$\boldsymbol{\varepsilon} \mid \sigma^2 \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$$

For simplicity, assume that σ^2 is fixed and that we put a prior on $\mathbf{w}$ which is:

$$\mathbf{w} \mid \mathbf{Q}^*, \sigma_0^2 \sim \mathcal{N}(\mathbf{0}, \sigma_0^2\mathbf{I})$$

The likelihood function is defined by the conditional distribution:

$$\mathbf{t} \mid \mathbf{Q}^*, \mathbf{w}, \sigma^2 \sim \mathcal{N}(\mathbf{Q}\mathbf{w}, \sigma^2\mathbf{I})$$

(a) (Adapted from FCML 3.9) Show that the posterior mean, $\mathbb{E}\left[\mathbf{w} \mid \mathbf{Q}^*, \mathbf{t}, \sigma^2, \sigma_0^2\right]$, is equivalent to the ridge regression solution:

$$\hat{\mathbf{w}}_\lambda = \left((\mathbf{Q}^*)^\mathsf{T}\mathbf{Q}^* + \lambda\mathbf{I}\right)^{-1}(\mathbf{Q}^*)^\mathsf{T}\mathbf{t}$$

for some λ. Find that λ in terms of σ^2, σ_0^2, N, and $\mathbf{Q}^*$ (all of which are given before we see the data $\mathbf{t}$).

(b) Find an expression for the log marginal likelihood:

$$\log p(\mathbf{t} \mid \mathbf{Q}^*, \sigma^2, \sigma_0^2) = \int p(\mathbf{t} \mid \mathbf{Q}^*, \sigma^2, \mathbf{w}) p(\mathbf{w} \mid \sigma_0^2)\, d\mathbf{w}$$

(c) Write a function that takes an $N \times (D+1)$ feature matrix $\mathbf{Q}^*$ (where the first column is assumed to be constant), a target vector $\mathbf{t}$, a "prior sample size" n_0, and an integer $d \in \{0, \ldots, D\}$, and does the following:

(i) Calculates $\hat{\sigma}^2$, the MLE for σ^2 based on a constant model for $\mathbf{t}$, i.e., using only the first column of $\mathbf{Q}^*$ as a predictor.

(ii) Holding $\sigma^2 = \hat{\sigma}^2$, sets $\sigma_0^2 = \sigma^2/n_0$, and calculates the log marginal likelihood for $\mathbf{t}$ using the first $d+1$ columns of $\mathbf{Q}^*$ as the predictor matrix.

(d) Using the `womens100` and `synthdata2016` data from HW1b, create a polynomial basis matrix $\mathbf{X}$ using polynomial degree $D = 9$, and use QR decomposition on this $\mathbf{X}$ to find $\mathbf{Q}^*$ (`numpy` has a function to do QR decomposition). Then use the function you wrote in the last step to get and graph the log marginal likelihood, for n_0 ranging logarithmically from 10^{-5} to 10^1.

(e) How does the choice of n_0 impact the optimal choice of polynomial order as measured by log marginal likelihood?