Homework 2
KNN, MLE, Naive Bayes

CMU 10-601: Machine Learning (Spring 2017)
https://piazza.com/cmu/spring2017/10601
OUT: February 1st
DUE: February 13th, 5:30 PM
Authors: Dylan, Edward, Brynn

START HERE: Instructions

- **Collaboration policy:** Collaboration on solving the homework is allowed, after you have thought about the problems on your own. It is also OK to get clarification (but not solutions) from books or online resources, again after you have thought about the problems on your own. There are two requirements: first, cite your collaborators fully and completely (e.g., "Jane explained to me what is asked in Question 3.4"). Second, write your solution *independently*: close the book and all of your notes, and send collaborators out of the room, so that the solution comes from you only. See the collaboration policy on the website for more information: `http://www.cs.cmu.edu/~mgormley/courses/10601-s17/about.html`
- **Late Submission Policy:** See the late submission policy here: `http://www.cs.cmu.edu/~mgormley/courses/10601-s17/about.html`
- **Submitting your work:** Each homework will consist of two parts. For this homework, the first part will be submitted via QnA. The second part of the assignment will be a programming question. For this question, you will submit code via Autolab and answer related questions on QnA as detailed below.
  - **QnA:** We will use an online system called QnA for short answer and multiple choice questions. You can log in with your Andrew ID and password using the button labeled "LOGIN WITH CMU ID". (As a reminder, never enter your Andrew password into any website unless you have first checked that the URL starts with "https://" and the domain name ends in ".cmu.edu" – but in this case it's OK since both conditions are met) A link to the Homework 2 QnA section is provided in Problem 1 below. The deadline displayed on QnA may not correspond to the actual deadline for this homework, since we are allowing late submissions (as discussed in the late submission policy on the course site).
  - **Autolab:** You can access the 10601 course on autolab by going to `https://autolab.andrew.cmu.edu/` Using All programming assignments will be graded automatically on Autolab using Octave 3.8.2 and Python 2.7. You may develop your code in your favorite IDE, but please make sure that it runs as expected on Octave 3.8.2 or Python 2.7 before submitting. The code which you write will be executed remotely against a suite of tests, and the results are used to automatically assign you a grade. To make sure your code executes correctly on our servers, you should avoid using libraries which are not present in the basic Octave install. For Python users, you are encouraged to use the `numpy` package. The deadline displayed on Autolab may not correspond to the actual deadline for this homework, since we are allowing late submissions (as discussed in the late submission policy on the course site).

Problem 1: QnA [50 points]

This problem consists of a set of multiple choice and numerical questions posted on the QnA platform at the URL below (also posted in Piazza):

https://qna.cs.cmu.edu/#/pages/view/112

The final set of questions on QnA are based on Problem 2 of this homework and should not be attempted until you have read through Problem 2 below.

QnA allows you to submit multiple times for the same question. We will grade the last answer you submit before the homework deadline.

Problem 2: Implementing Naïve Bayes [50 points]

For this question, you will implement a Naïve Bayes (NB) classifier. You are given a dataset containing text articles coming from two sources: *The Economist*, a serious news source, and *The Onion*, a sarcastic news source. You will train a classifier to distinguish between the articles from the two sources.

The features used to classify articles are generated from the words appearing in those articles. The set of all words from all the articles in our data is called the *vocabulary*, and let's say its size is V. We will represent each article as a feature vector $\mathbf{x} = \langle x_1, \ldots, x_V \rangle$, such that

$$x_w = \begin{cases} 1 & \text{if word } w \text{ is present in the document} \\ 0 & \text{otherwise} \end{cases}$$

We also associate each article with a label y such that

$$y = \begin{cases} 0 & \text{if the article is from } \textit{The Economist} \\ 1 & \text{if the article is from } \textit{The Onion} \end{cases}$$

We make two key assumptions with the Naïve Bayes classifier. First, we assume our data are drawn i.i.d from a joint probability distribution over feature vectors $\mathbf{X}$ and labels Y. More importantly, we assume that for each pair of features X_i and X_j with $i \neq j$, X_i is conditionally independent of X_j given the class label Y (hence the "naïve" in Naïve Bayes). To predict the label of an article, we choose the most probable class label given $\mathbf{X}$.

$$\hat{y} = \operatorname*{argmax}_y P(Y = y | \mathbf{X} = \mathbf{x})$$

Using the Bayes rule and the NB assumption, we can rewrite the above expression as follows

$$\begin{aligned} \hat{y} &= \operatorname*{argmax}_y \frac{P(\mathbf{X} = \mathbf{x} | Y = y) P(Y = y)}{P(\mathbf{x})} && \text{(Bayes' rule)} \\ &= \operatorname*{argmax}_y P(\mathbf{X} = \mathbf{x} | Y = y) P(Y = y) && \text{(denominator does not depend on } y\text{)} \\ &= \operatorname*{argmax}_y P(X_1 = x_w, \ldots, X_V = x_V | Y = y) P(Y = y) \\ &= \operatorname*{argmax}_y \prod_{w=1}^{V} P(X_w = x_w | Y = y) P(Y = y) && \text{(Naïve Bayes assumption)} \end{aligned}$$

By making the NB assumption, we can factor the probability distribution $P(\mathbf{X} = \mathbf{x} | Y = y)$ as a product of all $P(X_w = x_w | Y = y)$, allowing us to define and learn significantly fewer parameters.

1. **[2 points]** How many parameters will the model need under the NB assumption, assuming that $P(X_w = x_w|Y = y)$ and $P(Y = y)$ are both Bernoulli distributions? Give your answer as a function of the vocabulary size, V. Please submit your answer on QNA.

2. **[2 points]** How many parameters (also as a function of V) will the model need if we do not make the NB assumption, assuming $P(Y = y)$ is Bernoulli again and $P(\mathbf{X} = \mathbf{x}|Y = y)$ is a categorical distribution? Please submit your answer on QNA.

Since we do not know the true joint distribution over $\mathbf{X}$ and Y, we need to estimate $P(\mathbf{X} = \mathbf{x}|Y = y)$ and $P(Y = y)$ from the training data. For each word w and class label y, suppose that the distribution of X_w given Y is a Bernoulli distribution with the parameter θ_{yw}, such that

$$P(X_w = 1|Y = y) = \theta_{yw} \quad \text{and} \quad P(X_w = 0|Y = y) = 1 - \theta_{yw}$$

A common problem in language related ML problems is dealing with words not seen in training data. Without any prior information, the probability of unseen words is zero. We know that this is not a good estimate, and we want to assign a small probability to any word in the vocabulary occurring in either *The Economist* or *The Onion*. We can achieve that by imposing a Beta(α, β) prior on θ_{yw}, and perform a MAP estimate from the training data. The p.d.f. of the Beta distribution is given as follows:

$$f(\rho; \alpha, \beta) = \frac{1}{\mathbf{B}(\alpha, \beta)} \rho^{\alpha-1}(1-\rho)^{\beta-1} \ \forall y, w, \rho$$

where $\mathbf{B}(\alpha, \beta)$ is the beta function:

$$\mathbf{B}(\alpha, \beta) = \int_0^1 t^{\alpha-1}(1-t)^{\beta-1} dt$$

You will experiment with two combinations of α and β values in this problem. Assume the distribution of Y is a Bernoulli distribution (taking values 0 or 1), as given below.

$$P(Y = 0) = \phi \quad \text{and} \quad P(Y = 1) = 1 - \phi$$

Since we have enough articles in both classes, we need not worry about zero probabilities, and will not impose a prior on ϕ.

Programming Instructions

You will implement some functions for training and testing a Naïve Bayes classifier for this question. You will submit your code online through the CMU autolab system, which will execute it remotely against a suite of tests. Your grade will be automatically determined from the testing results. Since you get immediate feedback after submitting your code and you are allowed to submit as many different versions as you like (without any penalty), it is easy for you to check your code as you go.

To get started, you can log into the autolab website (`https://autolab.andrew.cmu.edu`). From there you should see 10-601 in your list of courses. Download the handout for Homework 2 (Options → Download handout) and extract the contents (i.e., by executing `tar xvf hw2.tar` at the command line). In the archive you will find three folders. The `data` folder contains the data files for this problem. The `python` folder contains a `NB.py` file which contains empty function templates for each of the functions you are asked to implement. Similarly, the `octave` folder contains separate `.m` files for each of the functions that you are asked to implement.

To finish each programming part of this problem, open the `NB.py` or the function-specific `.m` template files and complete the function(s) defined inside. When you are ready to submit your solutions, you will create a new tar archive of the files you are submitting the `hw2` directory. Please create the tar archive exactly as detailed below.

If you are submitting Python code:

```
tar cvf hw2_handin.tar NB.py
```

If you are submitting Octave code:

```
tar cvf hw2_handin.tar logProd.m NB_XGivenY.m NB_YPrior.m NB_Classify.m classificationError.m
```

If you are working in Octave and are missing **any** of the function-specific `.m` files in your tar archive, you will receive zero points.

We have provided all of the data for this assignment as `csv` files in the `data` folder in your handout. You can load the data using the `numpy.genfromtext` function in Python or the `csvread` and `csv2cell` (`io` package) functions in Octave. We have provided a `hw2_script` file for each language to help get you started. You should build on the `hw2_script` to test out the functions we are asking you to implement and turn in, but you will not submit the script itself. After loading the data, you should have 7 variables: `vocabulary`, `XTrain`, `yTrain`, `XTest`, `yTest`, `XTrainSmall`, and `yTrainSmall`.

- `vocabulary` is a $V \times 1$ dimensional vector that that contains every word appearing in the documents. When we refer to the j^{th} word, we mean `vocabulary(j)`. You will not need the vocabulary vector for any of the questions on this homework, but we have included it to help you better understand the data set.
- `XTrain` is a $n \times V$ dimensional matrix describing the n documents used for training your Naive Bayes classifier. The entry `XTrain(i,j)` is 1 if word j appears in the i^{th} training document and 0 otherwise.
- `yTrain` is a $n \times 1$ dimensional vector containing the class labels for the training documents. `yTrain(i)` is 0 if the i^{th} document belongs to The Economist and 1 if it belongs to The Onion.
- `XTest` and `yTest` are the same as `XTrain` and `yTrain`, except instead of having n rows, they have m rows. This is the data you will test your classifier on and it should not be used for training.
- Finally, `XTrainSmall` and `yTrainSmall` are subsets of `XTrain` and `yTrain` which are used in the final question.

Code

Logspace Arithmetic

When working with very large or very small numbers (such as probabilities), it is useful to work in *logspace* to avoid numerical precision issues. In logspace, we keep track of the logs of numbers, instead of the original values. For example, if $p(x)$ and $p(y)$ are probability values, instead of storing $p(x)$ and $p(y)$ and computing $p(x) * p(y)$, we work in log space by storing $\log(p(x))$, $\log(p(y))$, and we can compute the log of the product, $\log(p(x) * p(y))$ by taking the sum: $\log(p(x) * p(y)) = log(p(x)) + log(p(y))$.

3. **[4 points]** Complete the function `logProd(x)` which takes an input a vector of numbers in logspace (i.e., $x_i = \log p_i$) and returns the product of those numbers in logspace—i.e., `logProd(x)` $= \log\left(\prod_i p_i\right)$.

Training Naïve Bayes

4. **[6 points]** Complete the function `NB_XGivenY(XTrain, yTrain, alpha, beta)`. The output `D` is a $2 \times V$ matrix, where for any word w and class label y, the entry `D(y,w)` is the MAP estimate of $\theta_{yw} = P(X_w = 1|Y = y)$ with a Beta(α, β) prior distribution. A useful initial step will be to derive the closed form expression for the MAP estimate of Bernoulli parameter θ_{yw} with a Beta prior. Note

that the parameters of the Beta distribution are also given as input to the function.

5. **[6 points]** Complete the function `NB_YPrior(yTrain)`. The output `p` is the MLE for $\phi = P(Y = 0)$.

6. **[10 points]** Complete the function `NB_Classify(D, p, XTest)`. The input `XTest` is an $m \times V$ matrix containing m feature vectors (stored as its rows). D and p are in the same form of outputs of question 4 and 5. The output `yHat` is a $m \times 1$ vector of predicted class labels, where `yHat(i)` is the predicted label for the i^{th} row of `XTest`. [Hint: In this function, you will want to use the `logProd` function to avoid numerical problems.]

7. **[2 points]** Complete the function `classificationError(yHat, yTruth)`, which takes two vectors of equal length and returns the fraction of entries that disagree.

Experiments

8. **[6 points]** We will experiment with two different parameter settings for our prior over θ_{yw}:

   (a) $\alpha = 2$ and $\beta = 5$, and

   (b) $\alpha = 5$ and $\beta = 2$.

   For each of these settings, plot the probability density function for our prior over θ_{yw}. Which values of α and β make more sense if we strongly believe the true value of θ_{yw} lies in the interval $[0.1, 0.3]$? Please submit your answer on QNA. You do not need to submit your plots.

9. **[6 points]** Train your classifier on the data contained in `XTrain` and `yTrain` for both parameter settings in the previous problem. Then, use the learned classifier to predict the labels for the article feature vectors `XTest`. Based on the classification error on the test set, which parameter settings were a better choice for the prior on θ_{yw}? Please submit your answer on QNA.

10. **[6 points]** Now set $\alpha = 2$ and $\beta = 5$ as in the first experiment, but train your classifier using the smaller training set `XTrainSmall` and `yTrainSmall`. Report your test error on the test set on QNA. How does this compare to the test error from the larger data set with the same choice of α and β?