Instructions

- The homework is due on **Friday 2/11 at 5pm ET**.
- There are 2 problems in total.
- No extension will be provided, unless for serious documented reasons.
- **Start early!**
- Study the material taught in class, and feel free to do so in small groups, but the solutions should be a product of your own work.
- This is not a multiple choice homework; reasoning, and mathematical proofs are required before giving your final answer.
- The code necessary for problem 2 should be written in the Jupyter notebook handed out to you.

1 Bayes rule [30 points]

Let N be a discrete random variable that takes values from the set $\{1, n\}$ with equal probability, i.e., $\mathbf{Pr}[N = 1] = \mathbf{Pr}[N = n] = \frac{1}{2}$. Consider the following process.

- First we draw a value for N.
- Then, we draw N iid uniform RV $\{X_i\}_{i=1,\dots,N}$ in $[0, 1]$.

Someone tells you the value $Z = \min_{i=1,\dots,N} X_i = 0.05$, namely the smallest value among the N uniform RVs drawn. However you do not know the value of N.

What is the probability $\mathbf{Pr}[N = 1 | Z = 0.05]$ when: (a) $n = 2$, and (b) $n = 10$.

2 Naive Bayes Classifier [70 Points]

2.1 Overview/Task

The goal of this programming assignment is to build a naive Bayes classifier from scratch that can determine whether email text should be labeled as spam or "ham", where "ham" is "not spam" by looking at the probabilities associated with each word in the emails.

2.2 Review

Remember that a naive Bayes classifier realizes the following probabilities:

$$P(Y|X) \propto P(Y)P(X|Y)$$

$$P(Y|X) = \frac{P(Y)P(X|Y)}{P(X)}$$

Where Y is either 0 or 1 and X is the input we are attempting to classify.

The classifier will decide what class each input belongs to based on highest posterior probability, $P(Y|X)$, from the equations above.

Recall for a naive Bayes, when an input X has n features, x_0 through x_{n-1}:

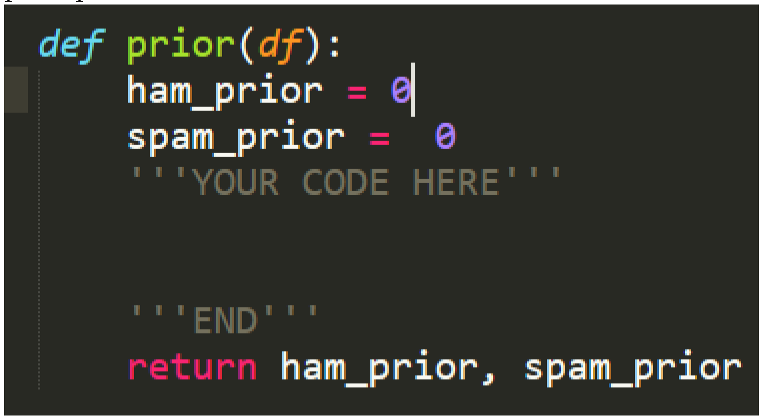
$$P(Y|x_0, ..., x_{n-1}) \propto P(Y) * P(x_1|Y) * P(x_2|Y) * ... * P(x_{n-1}|Y)$$

2.3 Requirements

1. Download the code template "PA2.py", the training file "TRAIN_balanced_ham_spam.csv", and the testing file "TEST_balanced_ham_spam.csv".

2. Rename the file to "Firstname_lastname_PA2.py"

3. Do not change any function names nor variable names that are **outside** of a coding prompts.



```
def prior(df):
    ham_prior = 0
    spam_prior =  0
    '''YOUR CODE HERE'''


    '''END'''
    return ham_prior, spam_prior
```

For instance, in the image above, you should NOT edit the name nor the parameters of the function "prior", the variable names "ham_prior" and "spam_prior", and the return variables of the same name

4. We have provided you with two data sets, training and testing. Each are in CSV format and should be loaded and explored as we have done previously in lab

2.4 Reminders

1. Please remember that the classifier must be written from scratch; do NOT use any libraries that implement the classifier for you, such as but not limited to SKlearn.

2. Feel free to look up any tasks you are not familiar with, e.g. the function call to read a CSV

2.5 Recommended task order

In order to provide some guidance, I am giving the following order/checklist to solve this task:

1. **10** points: Compute the "prior": P(Y) for Y = 0 and Y = 1
2. **20** points: Create a data structure to compute the "likelihood": $P(x_n|Y)$
3. **30** points: Write code that uses the two items above to make a decision on whether or not an email is spam or ham (aka not spam)
4. Test model on training data to debug. Should be getting an accuracy, precision, and recall of about : .735, .659, .9799, respectively.
5. Test model on testing data to debug. Should be getting an accuracy, precision, and recall of about: .7383, .6621, .9733, respectively