MATH 373: Intro to Machine Learning
Case Study 1: Spam Detection with Naïve Bayes

by James D. Wilson (University of San Francisco)

Directions: Provide all `R` code and solutions by *knitting* your final `R`Studio file into a single file. You can work in groups of up to 4, and only one person needs to upload the final .pdf to Canvas. *You must work in groups of at least size 2*! Be sure to put all of your group members' names at the top of the document.

1. Based on the lecture given in class about the Naïve Bayes classifier, I want you to build (from scratch) your own Naïve Bayes classification algorithm. Be sure to incorporate the following components to your algorithm:

   (a) **Input**:
   - an $n \times p$ data table or matrix X whose columns are *binary* variables and rows are observations
   - an n length response vector y, whose entries are factors representing labels for each observation

   (b) **Output**: a list containing the following objects:
   - a vector of prior probabilities for each label in y
   - class conditional probabilities $P(x_k = 1 \mid Y = j)$, for $k = 1, \ldots, p$ and all labels j
   - a vector of posterior probabilities $P(Y = j \mid X = x)$ calculated from your algorithm
   - an n dimensional vector specifying the classification for each observation
   - the misclassification rate (between 0 and 1) of your classifier

2. Now you will build a Naïve Bayes classifier on a collection of emails in the `Data/SPAMData` folder on the Github site. The dataset is the CSDMC2010 SPAM corpus, which was made publicly available for a data mining competition at the ICONIP conference in 2010. See the readme.txt file for more information. Perform the following steps:

   (a) Download the raw data in the `SPAMData` folder from Github and store it locally on your computer.

   (b) Extract the emails and labels of this data set by following along the example template `SpamEmailExtraction.R` file I created in the `Case Studies` folder on Github.

   (c) Now here's where you get to be creative. Come up with 20 or more binary features used to describe the email corpus that you think will be useful in detecting spam emails. I provided an example of how to count the number of instances of the word *explosive*. One could easily make this a binary feature for an email by setting a variable to 1 if the word *explosive* is in the text of the email and to 0 otherwise. Be as creative as you'd like to be here, and keep in mind that the aim is to come up with features that will distinguish spam from non-spam emails. This is called *feature engineering* by the way. Once you're done, create a matrix X using these binary features.

   (d) Based on the labels and the matrix X created above, run your Naïve Bayes classifier on the corpora of these emails. Report the misclassification rate of your classifier.

3. **Bonus**: for the above spam detection task, one can iterate between the creation of features, subsetting of features, and running the classifier. At the end of the day, you want the **best** classification rule to detect spam. The team that comes up with the features that result in the lowest misclassification rate on the full training data will receive +15 points to be spread across and added to Homework 0, Homework 1, and the first quiz. Good luck!