**Project 1: Email Classifier**

CS4981 ML Production Systems

**Overview**

In this lab, you are going to analyze 63,542 emails. You will convert the raw text into a feature matrix using a "bag of words" model. Each column of the feature matrix corresponds to one word, each row corresponds to one email, and the entry stores the number of times that word was found in that email. You will create and evaluate classifiers to distinguish between spam and ham emails based on vectorization of the email bodies.

**Instructions**

**Part I: Set Up a Local Python Environment**

We will be using Python 3, version 3.7 or later. If you are using a Mac or Linux environment, you will likely already have Python 3 installed. You can confirm this from the terminal like so:

$ python --version

Python 3.9.10

You may see Python 2.x listed like so:

$ python --version

Python 2.7.16

In that case, your system likely has both Python 2 and Python 3 installed. The Python executable will then be named python3:

$ python3 --version

Python 3.7.3

If you are using Windows, you will need to install Python and ensure that it is available in your path. You can [download installers](https://www.python.org/downloads/windows/) from the Python web site. The installer offers the option to add Python to your path automatically. You may need to open a new terminal session for that to take effect. Similarly, you can check that Python is installed and in your path from the command-line like so:

> python.exe --version

After the base Python language is installed, we will need to install a number of libraries. Python provides a feature called [“virtual environments”](https://packaging.python.org/en/latest/guides/installing-using-pip-and-virtual-environments/#creating-a-virtual-environment) that allow you to install libraries for a specific project in the user’s directories (rather than system wide). This feature makes it convenient to work on different code bases that need different or conflicted versions of the same libraries. To create a virtual environment, do the following:

$ python -m venv cs4981

where cs4981 is the name of the directory that will be created to hold the contents. Once created, you won’t need to create it again. If you make a mistake or when you finish the class, you can uninstall it simply by deleting the directory.

Whenever you want to use the virtual environment, you will need to activate it. You can do that like so:

$ source cs4981/bin/activate

or if on Windows:

> cs4981\Scripts\activate.bat

Your shell prompt will now change, and you confirm the location of the Python interpreter:

(cs4981) $ which python

/Users/rnowling/cs4981/bin/python

or if on Windows:

> where python

In your environment, you can now install the libraries we will need. Python provides a program called pip for downloading and installing libraries. We will start by upgrading pip and installing a package called wheel that allows your system to install pre-compiled versions of libraries.

(cs4981) $ pip install -U pip wheel

After that, install the various libraries we need:

(cs4981) $ pip install numpy scipy jupyter sklearn seaborn matplotlib pandas

I find PyCharm to be a particularly nice IDE. It allows pointing to a virtual environment, editing Python scripts, and working in Jupyter notebooks.

**Part II: Load the Data**

a. Download the email\_json\_dataset1.zip file from the shard Box folder and unzip it:

<https://msoe.box.com/s/l9xl4udcwvqzrzwtxwrqw4bejehomk2b>

b. Extract the zip file. It should create a directory called "email\_json" (or similar). Each email is stored as a separate JSON document with a name in the format "message\_XXXXX.json".

b. Write some code to find and load all of the JSON documents. You should have a list of dicts when done. (Hint: Review the built-in Python glob and json libraries.)

c. Convert the list of dicts into a Pandas DataFrame. The DataFrame should have a handful of columns such as from, to, subject, body, and label and 63,452 rows. What are the column names and their types? (Hint: use the DataFrame.from\_records() or DataFrame.from\_dict() functions.)

**Part III: Setup Experiment**

Design and apply an experimental setup (e.g., train-test split or cross-fold validation). Ensure that you handle class imbalance when training the models through down-sampling or up-sampling the training sets and choose metrics such as recall and precision that are appropriate for imbalanced data.

**Part IV: Extract Features**

By themselves, strings of the message bodies are not amenable to analysis. We need to convert them to a feature matrix.

a. Use Scikit-Learn's [CountVectorizer](https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html) class to create a feature matrix from the message bodies. The "bodies" column of the DataFrame can be used as a list of strings and passed directly into the fit\_transform() method of the CountVectorizer.

b. How many rows and columns does the feature matrix have? How many nonzero entries are in the matrix?

**Part V: Create and Evaluate a Simple Classifier**

Create a model that predicts whether an email is spam or not using the word count features as above. Evaluate several different model types (e.g., linear models, Random Forests, etc.). Perform parameter tuning on the CountVectorizer and classifier hyper-parameters. Evaluate the impact of class balance in the training. Report on the model accuracies and run times from training.

**Submission Instructions**

Save the Jupyter notebook as a PDF and upload that file to Canvas.