Spam/Ham Classification Project

This Project

In this project, you will use what you learned in class to create a classifier that can distinguish spam (junk or commercial or bulk emails) from ham (non-spam emails). In addition to providing some intuition (see in-lab 6).

Warning

This is **real world** dataset the emails you are trying to classify are actual spam and legitimate emails. As a result, some of the spam emails may be offensive or sexually explicit. We have attempted to minimize the amount of offensive and explicit content. If you encounter such content, we apologize for putting you in the position of having to look at such content.

Disclaimer about `sklearn` and `scikit-learn`

Throughout this project, we will be using `sklearn`, which is a popular machine learning package for `scikit-learn`. `sklearn` is written in C++ and is intended to be fast and performant, all `sklearn` code is in C, so you may have trouble in areas of the previous assignments, such as `numpy` broadcasting, being faster than `sklearn`. `sklearn` is also very well documented, so we encourage you to read the documentation, and the updated classes. Generally, `sklearn` is a great package to know for any future data science assignments or work involving

See the statistics documentation on distributions and functions for more details.

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
sns.set(style="whitegrid", palette="pastel", color_codes=True)
sns.set_context("notebook", rc={"figure.figsize": (8, 4)})
np.random.seed(42)
```

Part 1 - Initial Analysis

```
import os, shutil, math
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_regression import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
from scipy.stats import norm, t, chi2, f, binom, poisson, gamma, beta, uniform
```

Loading in the Data

In email classification, our goal is to classify emails as spam or not spam (referred to as "ham") using features generated from the text in the email.

The dataset consists of email messages and their labels (0 for ham, 1 for spam). The spam/ham dataset contains 5,695 labeled messages. 3,672 of the messages are confirmed ham and 1,018 are confirmed spam. The file containing the labels for the spam/ham messages is `spam_ham_labels.csv`. The messages are in the `spam_ham_data` folder.

Run the following cells to load in the data into DataFrames.

The `Subject` DataFrame contains 100 emails per fold (except fold 10, which has 95 emails). It contains four columns:

1. `id`: An identifier for the training example
2. `subject`: The subject of the email
3. `email`: The text of the email.
4. `spam`: 1 if the email is spam, 0 if the email is ham (not spam)

The `Tokens` DataFrame contains 100 emails per fold (except fold 10) and contains four columns:

1. `id`: An identifier for the training example
2. `email`: The text of the email.
3. `token`: A token or word, obtained by splitting the email on spaces.
4. `spam`: 1 if the email is spam, 0 if the email is ham (not spam)

```
!ls -lhtr data/spam_ham_data/
!head -10 data/spam_ham_data/EXPLANATION
```

```
total 1.0M
drwxr-xrx 2 dsc80 dsc80  4K Jun 11 00:00 .
drwxr-xrx 3 dsc80 dsc80  4K Jun 11 00:00 ..
-rwxr-xrx 1 dsc80 dsc80 1.0M Jun 11 00:00 EXPLANATION
-rwxr-xrx 1 dsc80 dsc80 1.0M Jun 11 00:00 EXPLANATION~
```

```
id
subject
email
spam
EXPLANATION
```

```
!head -1 data/spam_ham_data/EXPLANATION
```

```
id,subject,email,spam
```

```
!head -2 data/spam_ham_data/EXPLANATION
```

```
id,subject,email,spam
EXPLANATION
```

```
!head -3 data/spam_ham_data/EXPLANATION
```

```
id,subject,email,spam
EXPLANATION
EXPLANATION
```

```
!head -4 data/spam_ham_data/EXPLANATION
```

```
id,subject,email,spam
EXPLANATION
EXPLANATION
EXPLANATION
```

```
!head -5 data/spam_ham_data/EXPLANATION
```

```
id,subject,email,spam
EXPLANATION
EXPLANATION
EXPLANATION
EXPLANATION
```

```
!head -6 data/spam_ham_data/EXPLANATION
```

```
id,subject,email,spam
EXPLANATION
EXPLANATION
EXPLANATION
EXPLANATION
EXPLANATION
```

```
!head -7 data/spam_ham_data/EXPLANATION
```

```
id,subject,email,spam
EXPLANATION
EXPLANATION
EXPLANATION
EXPLANATION
EXPLANATION
EXPLANATION
```

```
!head -8 data/spam_ham_data/EXPLANATION
```

```
id,subject,email,spam
EXPLANATION
EXPLANATION
EXPLANATION
EXPLANATION
EXPLANATION
EXPLANATION
EXPLANATION
```

```
!head -9 data/spam_ham_data/EXPLANATION
```

```
id,subject,email,spam
EXPLANATION
EXPLANATION
EXPLANATION
EXPLANATION
EXPLANATION
EXPLANATION
EXPLANATION
EXPLANATION
```

```
!head -10 data/spam_ham_data/EXPLANATION
```

```
id,subject,email,spam
EXPLANATION
EXPLANATION
EXPLANATION
EXPLANATION
EXPLANATION
EXPLANATION
EXPLANATION
EXPLANATION
EXPLANATION
```

Question 1c

Discuss one thing you notice that is different between the two emails that might relate to the identification of spam.

Spam email contains more digits.

Training Validation Split

The data is already randomly ordered. We've chosen to have a separate 20-fold training/validation split **stratening** the emails that are not. We therefore suggest using folds 1-19 for training and folds 20 for validation. We will refer to folds 1-19 combined as the **training set** and fold 20 as the **validation set**. Leaving the performance of your classifier once you can finalize training, **Note that we can the email contents, since in 2021**. This will provide an unbiased estimate of how well your classifier will perform on unseen emails in the future. **We can modify this for the following questions, as we have depend on the contents of the emails.**

```
# Run this cell to load the email bodies into DataFrames.
train = sorted(glob.glob('data/spam_ham_data/*[1-9]/*'))
test = sorted(glob.glob('data/spam_ham_data/*0/*'))
```

```
!ls -lhtr data/spam_ham_data/
```

```
total 4K
drwxr-xrx 2 dsc80 dsc80  4K Jun 11 00:00 .
drwxr-xrx 3 dsc80 dsc80  4K Jun 11 00:00 ..
```

Basic Feature Engineering

We would like to take the text of an email and produce a feature vector that we can use to train. This is a classification problem, so we can use logistic regression to try to predict the probability that an email is spam, given its feature vector. We will start by using a simple feature vector. A good method of starting off on a feature engineering task is to brainstorm a list of features that you think might be useful. To address this, we can create several features derived from the email text and see how well they perform in our model.

Some ham and SPAM emails have between 0 and 10 consecutive exclamation marks. We will create a feature that counts the number of consecutive exclamation marks in a given email. Let's start by creating a function that takes in a string, finds all groups of consecutive exclamation marks in the string, and returns the length of the longest group.

Question 2

Create a function called `exclam_length` that takes in a string of `email` and a parameter `name` of `email`. It should return a single number, the length of the longest sequence of consecutive exclamation marks in `email`. If `email` contains no exclamation marks, it should return 0.

```
>>> exclam_length("Hello! Hi!", "hi!", "hello!")
>>> exclam_length("Wow!!! Hi!!", "hi!!!", "wow!!!")
```

```
array([1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0
```

```
C:\Users\Ryan\AppDataLocal\Temp\ipykernel_21464\1.py:2: FutureWarning: Downcasting object dtypes in .fillna(), .ffill(), .bfill() is deprecated and will
  Filled_Data[col] = Filled_Data[col].fillna(Filled_Data[col].mode()[0])
```