SMS Spam Classification

Leo Ding

Introduction:

In the technological era, people are constantly bombarded with messages whether it be text messages, emails, or the like. Generally, these communication methods are used between people who know one another, such as a family member, a friend, or even a coworker. Every once in a while, though, people receive random messages from people they have never spoken to in their life. These are spam messages; these messages have been known to either be used as an advertisement method for different groups or a scamming attack.

Regardless of which type of spam the message is, most people would rather not have to filter through their own messages to find which ones are legitimate and which one is trying to give them $1000 simply by clicking a link and entering one's financial information. Thus, spam filters were created in order to automatically find a spam message and make sure that the person doesn't receive said message.

In this project, I set out to create a machine learning model to classify SMS messages as either spam or ham (not spam).

Dataset:

The dataset was acquired from UCI's Machine Learning Repository. The data is simply a file which contains 5574 different SMS message strings, with each message being marked as 'ham' or 'spam' before the actual message.

Method and Analysis:

After reading in the data file, each line was split up into two pieces and inserted into their own respective arrays. The first array kept track of the message markers, ham or spam, and converted them to binary values of 0 and 1 respectively. The second array held the actual string of the messages. While adding the strings to the array, select special characters and extra whitespace were removed.

Next, a TfidfVectorizer (Term Frequency Inverse Document Frequency) object was used to determine the importance of all of the words found throughout all of the messages. A matrix is returned with a row for each message, a column for each word found, and values were placed in cells to determine a word's importance in a certain sentence. The vectorized set of sentences was then rejoined with their spam markers, in order to be properly split into a spam and ham array.

Now that the spam and ham are separated, we can create train test splits per array (75% train, 25% test). Once train test splits are created for both spam and ham, we can recombine the sets into general train test splits. This is done to ensure that the proportion of spam messages in the train test split sets is similar to the proportion found in the original dataset.

Lastly, a logistic regression model was trained and tested on the datasets from above.

Results:

After running the logistic regression model, scikit-learn metrics were used to calculate accuracy, precision, and recall scores. The accuracy came out to be 0.9684 which is quite high, but it doesn't tell us too much extra information about how good the model is, thus, precision and recall were also calculated. Note that spam messages are considered positive values while ham messages are considered negative values. The model's precision and recall were 1 and 0.7647 respectively. The precision of 1 denotes that the model was able to not misclassify any ham messages (false positive) which is great since all of legitimate messages would have made it through to the user. The recall of 0.7647 denotes that the model, about 1/4 of the spam messages were marked as ham messages which can be considered a poor outcome. Even though spam messages were let through the filter, it is arguably more important for all of the legitimate messages to pass through. In the end, this model would cause the user to need to comb through less spam, which is a positive.