Spam Classification for SMS Messages

# **Ali Malenchik DSC 680 T301-2221 Fall 2021 https://github.com/alimalenchik/Portfolio**

# Which Domain?

This data comes from the personal data domain.

References:

* <http://cs229.stanford.edu/proj2013/ShiraniMehr-SMSSpamDetectionUsingMachineLearningApproach.pdf>
  + This whitepaper explores a spam detection data science project using Naïve Bayes, SVM, k-NN, Random Forest, and Adaboost algorithms.
* <https://towardsdatascience.com/the-ultimate-guide-to-sms-spam-or-ham-detector-aec467aecd85>
  + This blog walks through the theoretical AI concept for spam classifiers as well as data preparation, exploratory data analysis, and multiple classification algorithms. It also explores performance measurement criterion.
* <https://towardsdatascience.com/email-spam-detection-1-2-b0e06a5c0472>
  + This article explains the process of designing a spam filtering system using count vectorization and word embedding for feature extraction. It also explores scoring metrics and weighs the advantages and disadvantages of each metric.
* <https://towardsdatascience.com/nlp-spam-detection-in-sms-text-data-using-deep-learning-b8632db85cc8>
  + This blog explains the process of implementing Dense, Long Short Term Memory (LSTM), and Bidirectional-LSTM deep learning models to build an SMS spam detection model.
* <https://www.enjoyalgorithms.com/blog/email-spam-and-non-spam-filtering-using-machine-learning>
  + This article describes the steps to implement a email spam classifier using k-NN and discusses how real-world companies such as Gmail, Outlook, and Yahoo detect spam.
* <https://www.analyticsvidhya.com/blog/2021/08/email-spam-detection-a-comparative-analysis-of-4-machine-learning-models/>
  + This blog analyzes the difference between four machine learning models for spam detection using Python.
* <https://iopscience.iop.org/article/10.1088/1742-6596/1797/1/012017/pdf>
  + This journal illustrates the architectural workflow for implementing an SMS spam classifier, from data collection, to data cleaning, all the way to predictions.
* <https://www.kdnuggets.com/2017/03/email-spam-filtering-an-implementation-with-python-and-scikit-learn.html>
  + This blog walks through the feature extraction process for unstructured text, including removing stop words, lemmatization, etc.
* <https://analyticsindiamag.com/hands-on-guide-to-detecting-sms-spam-using-natural-language-processing/>
  + This guide explores a Python implementation of an SMS spam detection model using Natural Language Processing (NLP). It provides sample code for creating a corpus, creating word clouds, converting text to vectors, and more.
* <https://www.kaggle.com/dktalaicha/sms-spam-detection-with-nlp>
  + This sample project performs exploratory data analysis and builds a bag-of-words model on SMS spam detection data.

# Which Data?

The dataset I will be examining is a collection of 5572 SMS messages that have been collected for mobile phone spam research, found on [Kaggle](https://www.kaggle.com/team-ai/spam-text-message-classification). The dataset is provided in CSV format and contains two columns: Category and Message. The Message column contains the raw SMS message, while the Category column is the binary target variable classifying the message as spam or not. Possible Category values are “spam” or “ham” (not spam). The messages were gathered from multiple sources:

* The Grumbletext website, which is a UK forum for cell phone users to report received spam messages
* NUS SMS Corpus from the Department of Computer Science at the National University of Singapore
* A PhD Thesis created by Caroline Tag
* SMS Spam Corpus v.0.1 Big

# Research Questions? Benefits? Why analyze these data?

Spam detection is a valuable tool used to detect and prevent unsolicited and unwanted messages. SMS spam typically involves bulk messaging for commercial advertising or phishing links. Since sending spam is relatively cheap for spammers, it’s an easily abused method of communicating business interest. Many recipients believe spam to be annoying or a violation of privacy, and consequently a method of detecting and reducing spam is a meaningful way to improve the user experience.

My research questions are as follows:

* What is the distribution of spam vs. not spam messages in the dataset? Is the target variable balanced?
* Is there any observable pattern to the spam messages? Is there a difference in content of spam vs. not spam messages?
* Which algorithm performs best in classifying the messages as spam vs. not spam?
* Which performance metric should be used to evaluate the performance of the models?
* Using the best performing model, what percentage of spam messages can be detected? What percentage of legitimate messages would be lost?

# What Method?

To begin, the data will be split into a test and train dataset in order to reserve unseen data for model testing and evaluation. During exploratory data analysis, the distribution of the target variable will be represented visually using a bar chart. This will allow us to understand if the target variable is balanced. If the target variable is imbalanced, steps will be taken to correct this by over- or under-sampling.

Next, the data will be cleaned and prepared using Natural Language Processing (NLP). Stopwords will be removed, and tokenization and vectorization will be performed on all messages. The distributions of metrics such as number of words, frequently used top-used words, usage of non-alpha characters and symbols, etc. will be plotted for comparison. Word clouds will be created to allow readers to briefly understand high level differences between the messages.

Once exploratory data analysis is complete, multiple classification models will be trained on the training dataset to determine the best performing algorithm, including Random Forest, Naïve Bayes, and Decision Tree. Each model will be evaluated to determine the “best” performing. In the case of spam classification, it is imperative that non-spam messages aren’t inaccurately labeled as such. In other words, the ideal classification model would have the fewest false positives (messages predicted as “spam” by the model which are actually “ham”). Based on this, precision may be the best metric for evaluating the models as it calculates the ratio of true positives to predicted positives. Finally, a confusion matrix will be created in order to summarize the performance of the model.

# Potential Issues?

I may run into issues tokenizing and vectorizing the data, as I don’t have much experience with Natural Language Processing and it has been some time since I cleaned and perform feature extraction from unstructured text. I also anticipate having difficulty determining which hyperparameters and values to select during model tuning.

# Concluding Remarks

Spam is a major issue in mobile communication today. SMS messages sent to users without their prior permission are not only an annoyance but also may be a security concern. Machine learning classifiers offer a solution to detect and filter out spam, but the behavior of the filter must be carefully scrutinized to avoid losing legitimate messages. Different types of classifiers (Random Forest, Naïve Bayes, and Decision Tree) are tested in order to compare the performance and determine the most suitable approach for categorizing the SMS messages as “spam” or “ham”.