1. Problem Description

Spam is unwanted and unsolicited messages sent electronically. These spam messages often have malicious intent, and range from misleading advertising to phishing and malware spreads. Thus, spam is detrimental to both users and services, and creates mistrust and wariness between the two parties.

Furthermore, spam is rapidly on the rise, with the Federal Trade Commision reporting $8.8 billion in total reported losses in 2022, compared to the $6.1 billion in 2021 and the mere $1.2 billion in 2020 (FTC.gov). Therefore, it is increasingly important for companies and services to detect and filter spam messages.

My capstone project aims to ***create an ML model that is capable of detecting spam email messages***.

2. Data

My data **(**Data.csv) includes 39,513 email entries, with 20,445 labeled as ham and 19,068 as spam. Note: the "ham" label means that the message was legitimate.

The dataset contains the following columns:

Column	Description
Subject	The subject line of the email
Message	The content of the e-mail. Can contain an empty string if the message had only a subject line and no body. In case of forwarded emails or replies, this also contains the original message with subject line, "from:", "to:", etc.
Label	Whether the email was spam (1) or ham (0)

This dataset is made up of two premade datasets:

- SpamAssassin dataset
- Enron Spam dataset

Further details about these two datasets can be found at: DataCollection.md. Furthermore, DataCollection.py contains the script for processing the SpamAssassin dataset, as well as merging it with the Enron Spam dataset before outputting it as Data.csv.

3. Outline of Approach:

Given an email message as input, my Capstone Project will aim to classify the email as spam or ham. Since my training data is labeled, this will require a supervised classification model.

While implementing this ML model, I want to play around with both “traditional” machine learning approaches (such as Random Forest, Decision Tree, XGBoost, Naive Bayes, etc) as well as a deep learning approach (such as a neural network).

4. Final deliverable

My final deliverable will be an application deployed as a web service with an API.

5. Computational Resources Needed:

After discussing with my mentor, we have decided that my data is small enough to be able to run my model in Google Colab using either GPU or TPU.