# Spam Email Filtering Proposal with Machine Learning Classifiers

## 1. Introduction

Spam emails that contain phishing techniques threaten corporate data security as it leads to data breaches that devastate a company's finances and reputation *(Acronyms, 2025)*. These emails impersonate well known figures to deceive recipients into revealing financial data and login credentials *(Vailmail, 2024)* to exploit and expose sensitive information. In 2025, an estimation of 3.4 billion spam emails are sent to individuals daily *(AAG, 2025)* with 50 million phishing attacks being successful. This showcases the necessity of addressing spam email risks to improve data security within companies.

It is proposed to implement a machine learning model that successfully filters and detects spam emails through classification. The primary objective of this project is to train a classifier using a preprocessed dataset, enabling it to extract and analyse key features that assists the model to identify spam emails through pattern recognition. The chosen classifier for this project is the Random Forest Classifier.

## 2. Investigation of Classifier

### 2.1 Random Forest Classifier

#### 2.1.1 Method description

The Random Forest Classifier is a machine learning algorithm that creates multiple decision trees and merges them together to formulate an accurate and stable prediction *(Builtin, 2024)*. This method is effective with differentiating between legitimate and illegitimate emails through its approach of selecting relevant features *(Murti, 2023)* from within a random subset, introducing diversity among the decision trees and reducing overfitting risks and bias. This enables the machine learning model to achieve a higher performance accuracy *(Hu, 2022)* as the diversity ensures that each decision tree encompasses an aspect from within the entire dataset.

Furthermore, the decision trees are formed using the Gini Index, a metric that measures the probability of a feature being misclassified when selected randomly *(Murti, 2023)*. It is calculated through this equation as seen in figure 1,

$$\text{GINI INDEX} = 1 - \sum_{i=1}^{n}(Pi))^2$$
$$= 1 - [(P+)^2 + (P-)^2]$$

*Figure 1 - Gini Index formula*

and it guides the decision tree on the best path when splitting the dataset at each node. The Random Forest Classifier aims to achieve a low Gini Index, a result where the decision trees are split in pure subsets. This reduces the risk of misclassifications *(Builtin, 2024)*, enabling the classifier to perform reliable and clearer decisions. However, despite this classifier's effectiveness, it has the limitation of being too computationally intensive *(Murti, 2023)* with large datasets, due to its requirement of constructing multiple decision trees to formulate a result. Thus, the Random Forest Classifier is an effective method for differentiating between legitimate and illegitimate emails despite its limitations.

### 2.1.2 Key components

In exploring the process of implementing a Random Forest classifier to differentiate between legitimate and illegitimate emails, JupyterLab was utilised to train and test the model:

1. Import the necessary libraries and modules

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import string
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
import nltk
from nltk.corpus import stopwords
```

*Figure 2 - Libraries and modules*

2. Load the spam.csv dataset

```
df = pd.read_csv('spam.csv', encoding='latin-1')
df
```

*Figure 2.1.x - Converting csv file contents into Pandas DataFrame*

| | v1 | v2 | Unnamed: 2 | Unnamed: 3 | Unnamed: 4 |
|---|---|---|---|---|---|
| **0** | ham | Go until jurong point, crazy.. Available only ... | NaN | NaN | NaN |
| **1** | ham | Ok lar... Joking wif u oni... | NaN | NaN | NaN |
| **2** | spam | Free entry in 2 a wkly comp to win FA Cup fina... | NaN | NaN | NaN |
| **3** | ham | U dun say so early hor... U c already then say... | NaN | NaN | NaN |
| **4** | ham | Nah I don't think he goes to usf, he lives aro... | NaN | NaN | NaN |
| **...** | ... | ... | ... | ... | ... |
| **5567** | spam | This is the 2nd time we have tried 2 contact u... | NaN | NaN | NaN |
| **5568** | ham | Will ì_ b going to esplanade fr home? | NaN | NaN | NaN |
| **5569** | ham | Pity, * was in mood for that. So...any other s... | NaN | NaN | NaN |
| **5570** | ham | The guy did some bitching but I acted like i'd... | NaN | NaN | NaN |
| **5571** | ham | Rofl. Its true to its name | NaN | NaN | NaN |

5572 rows × 5 columns

*Figure 3 - Unedited table*

3. Drop redundant columns and rename them appropriately

```
df = df.drop(['Unnamed: 2','Unnamed: 3','Unnamed: 4'],axis=1)
df = df.rename(columns = {'v1':'label','v2':'message'})
```

*Figure 4 - Dropping and renaming columns*

| | label | message |
|---|---|---|
| **0** | ham | Go until jurong point, crazy.. Available only ... |
| **1** | ham | Ok lar... Joking wif u oni... |
| **2** | spam | Free entry in 2 a wkly comp to win FA Cup fina... |
| **3** | ham | U dun say so early hor... U c already then say... |
| **4** | ham | Nah I don't think he goes to usf, he lives aro... |
| **...** | ... | ... |
| **5567** | spam | This is the 2nd time we have tried 2 contact u... |
| **5568** | ham | Will ì_ b going to esplanade fr home? |
| **5569** | ham | Pity, * was in mood for that. So...any other s... |
| **5570** | ham | The guy did some bitching but I acted like i'd... |
| **5571** | ham | Rofl. Its true to its name |

5572 rows × 2 columns

*Figure 5 - Edited table*

4. Preprocess the text

```
def text_process(text):

    text = text.translate(str.maketrans('', '', string.punctuation))
    text = [word for word in text.split() if word.lower() not in stopwords.words('english')]

    return " ".join(text)

text_feat = text_feat.apply(text_process)
```

*Figure 6 - Removing punctuation and stopwords*

5. Extract dataset features using (TF-IDF Vectorization)

```
vectorizer = TfidfVectorizer()
features = vectorizer.fit_transform(text_feat)
```

*Figure 7 - Converting textual data into numerical format*

6. Split dataset for model training and testing

```
features_train, features_test, labels_train, labels_test = train_test_split(features, df['label'], test_size=0.2, random_state=42)
```

*Figure 8 - 80% of the dataset for model training and 20% for testing (test size is set to 0.2)*

7. Train the model using the training data

```
clf = RandomForestClassifier(n_jobs = -1)
clf.fit(features_train, labels_train)
```

```
[137]:
```

```
RandomForestClassifier
RandomForestClassifier(n_jobs=-1)
```

*Figure 9 - Using all available cores for computation*

### 2.1.3 Test Data

**Identified legitimate email:**

1. Load the chosen email

```
email_to_classify = df.message.values[4]
email_to_classify
```

```
[179]:

"Nah I don't think he goes to usf, he lives around here though"
```

*Figure 10 - loaded email #4*

2. Classify the chosen email

```
email_text = email_to_classify.lower().translate(str.maketrans('', '', string.punctuation)).split()
email_text = ' '.join(email_text)
email_corpus = [email_text]
x_email = vectorizer.transform(email_corpus)
clf.predict(x_email)
```

```
array(['ham'], dtype=object)
```

*Figure 11 - Successfully identified as ham*

**Identified illegitimate email:**

1. Load the chosen email

```
email_to_classify = df.message.values[2]
email_to_classify
```

```
[187]:

"Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text
FA to 87121 to receive entry question(std txt rate)T&C's apply 08452810075
over18's"
```

*Figure 12 - Loaded email #2*

2. Classify the chosen email

```
email_text = email_to_classify.lower().translate(str.maketrans('', '', string.punctuation)).split()
email_text = ' '.join(email_text)
email_corpus = [email_text]
x_email = vectorizer.transform(email_corpus)
clf.predict(x_email)
```

```
array(['spam'], dtype=object)
```

*Figure 13 - Successfully identified as spam*

### 2.1.4 Metrics and Evaluation

To ensure the Random Forest Classifier's effectiveness with spam filtering and detection, its 97.30% accuracy score was identified as seen in figure 14. This showcases the classifier's reliability with correctly labeling legitimate and illegitimate emails through classification. In figure

15, further evidence of this classifier's accuracy can be distinguished through its 0% false positive rate as there were no legitimate emails incorrectly labeled. However, there were 32 spam emails that were incorrectly classified as legitimate, resulting in a 21.33% false negative rate that showcases the possibility of the model being underfitted and requiring more training from the provided dataset. Thus, it is recommended to utilise more feature extraction techniques in the future to further improve this model's accuracy.

```
clf.score(features_test, labels_test)
```

```
0.9730861244019139
```

*Figure 14 - Accuracy score*

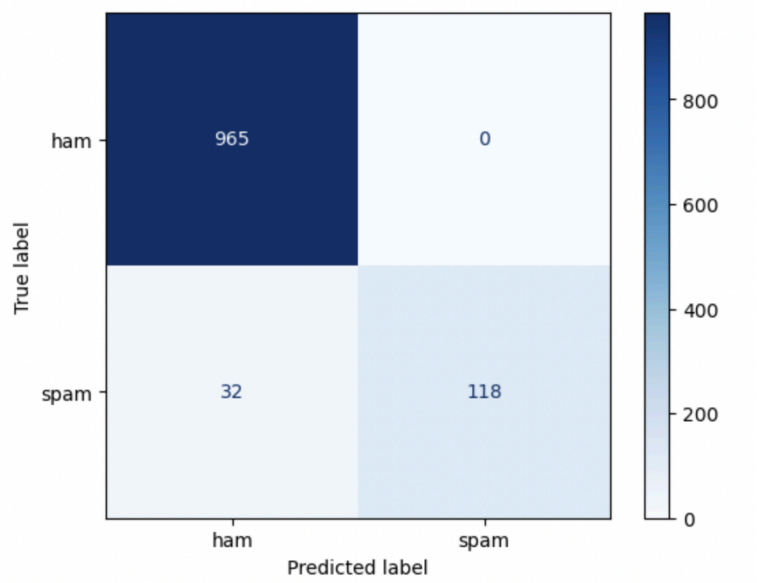


*Figure 15 - Confusion matrix for model with 0.2 test size*

# 6. References


1. AAG. (2025). *The Latest 2025 Phishing Statistics (updated January 2025).* https://aag-it.com/the-latest-phishing-statistics/
2. Acronyms. (2025). *The Dangers of Spam Email and How to Avoid Receiving it.* https://www.acronyms.co.uk/blog/dangers-of-spam-email/#:~:text=For%20individuals%2C%20the%20risks%20include,losses%20and%20lasting%20reputational%20damage.
3. Albe, Bee Theng Lau, Kit, M., & McCarthy, C. (2024). Enhancing road safety with machine learning: Current advances and future directions in accident prediction using non-visual data. *Engineering Applications of Artificial Intelligence*, *137*, 109086–109086. https://doi.org/10.1016/j.engappai.2024.109086
4. Builtin. (2024). *Random Forest: A Complete Guide for Machine Learning*. https://builtin.com/data-science/random-forest-algorithm
5. D'Souza, J. (2018, April 3). An Introduction to Bag-of-Words in NLP. Medium; GreyAtom. https://medium.com/greyatom/an-introduction-to-bag-of-words-in-nlp-ac967d43b428
6. Hashemi-Pour, C. (2023, December). *What is the CIA Triad? Definition, Explanation and Examples*. TechTarget. https://www.techtarget.com/whatis/definition/Confidentiality-integrity-and-availability-CIA
7. Murti, Y. S., Naveen, P. (2023). Machine Learning Algorithms for Phishing Email Detection. *Journal of Logistics, Informatics and Service Science.* Vol. 10 (2023) No.2, pp.249-261. https://www.aasmr.org/liss/Vol.10/No.2%202023/Vol.10%20No.2.17.pdf
8. Samy Baladram. (2024, August 20). *K Nearest Neighbor Classifier | TDS Archive*. Medium; TDS Archive.

https://medium.com/data-science/k-nearest-neighbor-classifier-explained-a-visual-guide-with-code-examples-for-beginners-a3d85cad00e1

9. SentinelOne. (2025, January 20). *What is Email Security and Why is it Important?* SentinelOne. https://www.sentinelone.com/cybersecurity-101/threat-intelligence/what-is-email-security/