Phishing and Spam Email Classification Using Machine Learning Algorithms

---

1. Problem Statement

Email is a primary medium for phishing and spam attacks that deceive users into sharing personal or financial information.
Traditional rule-based filters are ineffective against new and sophisticated attacks.
The goal of this project is to **develop a machine learning model** that automatically classifies emails as *legitimate, spam, or phishing* based on their content and metadata.

2. Methodology Overview

Here's a typical research pipeline:

Step 1 — Data Collection

Use publicly available datasets:

- **Spam Assassin Public Corpus**
- **Enron Email Dataset**
- **Phishing Email Dataset (UCI Repository)**
- **Kaggle Datasets** such as "Phishing Email Detection" or "Spam Text Message Classification"

You can also combine spam + phishing datasets to create a 3-class classification problem.

Step 2 — Data Pre-processing

Emails need cleaning before model training:

- Remove HTML tags, URLs, and special characters
- Convert to lowercase
- Tokenize and remove stop words
- Perform **stemming/lemmatization**
- Extract useful **features** such as:
    - Email subject and body text
    - Presence of links or suspicious keywords
    - Sender domain reputation
    - Number of attachments

Use Python libraries like:

```
nltk, scikit-learn, pandas, re, Beautiful Soup
```

Step 3 — Feature Extraction

Convert text into numerical form:

- **Bag of Words (BoW)**
- **TF-IDF (Term Frequency–Inverse Document Frequency)**
- **Word Embedding's (Word2Vec, Glove, or BERT)** for advanced models.

Example in Python:

```
from sklearn.feature_extraction.text import TfidfVectorizer
vectorizer = TfidfVectorizer(max_features=5000)
X = vectorizer.fit_transform (email_texts)
```

Step 4 — Model Building (Algorithms)

Try and compare multiple ML algorithms:

Algorithm	Why Use It	Difficulty
Naïve Bayes (MultinomialNB)	Fast, works well for text	⭐ Easiest
Logistic Regression	Simple baseline, interpretable	⭐
Support Vector Machine (SVM)	High accuracy for small datasets	⭐ ⭐
Random Forest	Handles non-linear patterns	⭐ ⭐
XGBoost / LightGBM	Great performance	⭐ ⭐ ⭐
Deep Learning (LSTM/BERT)	Best accuracy, complex	⭐ ⭐ ⭐ ⭐

Start simple with **Naïve Bayes** and **SVM**, then try a **deep learning** model if you want to extend it.

Step 5 — Model Training & Evaluation

Split your data:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
```

Train model and evaluate:

```
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report

model = MultinomialNB()
model.fit(X_train, y_train)
```

```
y_pred = model.predict(X_test)

print(accuracy_score(y_test, y_pred))
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

Key metrics to report:

- **Accuracy**
- **Precision**
- **Recall**
- **F1-score**
- **Confusion matrix**
- **ROC-AUC curve**

Step 6 — Optional Enhancements (for Research Depth)

If you want to make it PhD-level publishable:

1. **Explainable AI (XAI):** Use SHAP or LIME to show which words/features lead to classification decisions.
2. **Hybrid Approach:** Combine NLP + email header metadata (sender, domain age, link count, etc.).
3. **Adversarial Email Detection:** Test robustness against slightly modified phishing emails.
4. **Ensemble Models:** Combine multiple classifiers for higher accuracy.
5. **Online Learning Model:** Real-time update as new email data arrives.

3. Tools & Technologies

- **Languages:** Python
- **Libraries:** scikit-learn, pandas, numpy, nltk, matplotlib, seaborn
- **Advanced NLP:** TensorFlow / PyTorch (for deep learning), transformers (for BERT)
- **Visualization:** seaborn, plotly, SHAP
- **IDE:** Jupyter Notebook, VS Code

4. Expected Results

- Accuracy: 90–98% (depending on algorithm and dataset)
- Better detection for phishing than simple keyword filters
- Explainable model insights (why an email was flagged)

5. Possible Research Contributions

- A comparative analysis of ML algorithms for phishing detection
- A hybrid phishing detection model using NLP + metadata features
- Explainable phishing detection for end-user trust

- Low-resource model for real-time email classification

6. References (Datasets & Papers)

- UCI Machine Learning Repository — *Phishing Websites Dataset*
- Kaggle — *Phishing Email Detection Dataset*
- Paper: “A Comparative Study on Email Spam Classifications using Machine Learning Techniques” (IEEE, 2023)
- Paper: “Hybrid Machine Learning Approach for Phishing Detection in Emails” (Elsevier, 2022)