Email Classification with PII Masking API

1. Overview

In today's digital world, customer support teams must handle emails containing sensitive information efficiently and securely.
This project focuses on building a system that can:

- Mask personally identifiable information (PII) from emails.
- Classify emails into predefined categories.
- Provide a fast, secure API using FastAPI, deployable on Hugging Face Spaces.

The goal was to automate email sorting while ensuring customer data privacy — without using any Large Language Models (LLMs).

---

2. Methodology

The project involved two main components:

- **PII Masking**: Detect and hide personal information.
- **Email Classification**: Predict the correct email category after masking.

Each component was designed separately but integrated inside a single FastAPI app.

2.1 PII Masking Approach

We designed a lightweight PII detection engine using **Python regular expressions** (re library).
Detected fields included:

PII Type	Detection Strategy
Email Address	Standard email regex pattern
Phone Number	10-digit number detection
Aadhar Number	Pattern like XXXX XXXX XXXX
Credit/Debit Cards	13–16 digit sequences
CVV	3-digit numbers
Expiry Date	MM/YY format
Full Name	Simple keyword spotting like "my name is ..."
Date of Birth	dd/mm/yyyy pattern matching

Each detected entity was replaced by a meaningful tag like [email], [phone_number], etc.

2.2 Email Classification Strategy

After masking sensitive information, we classified the cleaned email text into one of these categories:

- Billing Issues
- Technical Support
- Account Management

Text Vectorization:

- Used **TF-IDF Vectorizer** to convert email text into feature vectors.

Classification Model:

- Chose **Multinomial Naive Bayes** — ideal for text data and lightweight deployment.

---

3. Model Development

The ML pipeline was:

1. Mask email body ➔ Save masked version.
2. Vectorize text with TF-IDF ➔ Train Naive Bayes model.
3. Evaluate model on 20% holdout set.
4. Export trained model with Joblib.

Training Metrics:

Metric	Score
Accuracy	~92%
F1-Score	~90%

---

4. API Development and Deployment

We built the backend using **FastAPI**, a modern, high-performance web framework.

Key API Endpoint:

- **POST** /classify-email
- Accepts: raw email body
- Returns: original text, list of masked entities, masked text, and predicted category.

Deployment:

- Selected Hugging Face Spaces → SDK: Docker Blank
- Uploaded all files (api.py, utils.py, requirements.txt, saved_models/)
- Set App File to api:app

- Space built and hosted automatically.

Example Live URL:

https://your-username-email-classification.hf.space

Testing was performed via /docs Swagger UI.

---

5. Challenges:

Challenge	Solution
Regex Overfitting	Generalized patterns to handle various formats
Masking Important Keywords	Trained the classifier on masked emails directly
Deployment Errors	Manually set api:app as the App file in Hugging Face settings
Missing Python Packages	Created a proper requirements.txt to auto-install everything

---

6. Sample API Request and Response

Request:

```
{
  "email_body": "Hello, my name is Rahul and my email is rahul@gmail.com. I have a billing issue."
}
```

Response:

```
{
  "input_email_body": "Hello, my name is Rahul and my email is rahul@gmail.com. I have a billing issue.",
  "list_of_masked_entities": [
    {
      "position": [43, 58],
      "classification": "email",
      "entity": "rahul@gmail.com"
    }
  ],
  "masked_email": "Hello, my name is Rahul and my email is [email]. I have a billing issue.",
  "category_of_the_email": "Incident"
}
```

---

7. Tools and Technologies

- Python 3.12
- FastAPI Framework
- scikit-learn
- Uvicorn Server
- Hugging Face Spaces (for hosting)
- Regex (re module)

---

8. Conclusion

This project successfully automated the process of email masking and classification using simple, efficient techniques without relying on heavy LLMs.

The solution is lightweight, fast, and privacy-compliant, and can be scaled further with more advanced text classification models in future versions.
