Project Report:

Email Classification API with PII Masking

API Link : https://rsrlearner-emailclassifiermaskeddata.hf.space/classify_email

1. Introduction

In today's customer support systems, large volumes of emails are received daily, covering a wide range of topics such as billing issues, technical support, account access, and more. Efficiently categorizing these emails can help route them to the appropriate department, reducing response time and improving user satisfaction.

Simultaneously, it is critical to ensure the privacy of customer data by masking Personally Identifiable Information (PII) such as names, emails, phone numbers, and credit card information. This is especially important when storing data for training or sharing logs with development teams.

The goal of this project is to build a FastAPI-based RESTful service that:
- Automatically classifies support emails into predefined categories
- Masks PII using regular expressions and NLP techniques
- Can be deployed both locally and on Hugging Face Spaces

2. Approach

2.1 PII Masking

To anonymize sensitive data, we designed a PII masking utility that uses a two-stage approach:

1. Pattern-based matching (Regex):
   Regular expressions were used to detect and mask:
   - Credit/debit card numbers
   - Aadhar numbers
   - Email addresses
   - Phone numbers
   - CVV and expiry dates
   - Dates of birth

2. Named Entity Recognition (NER) using spaCy:
We used spaCy's en_core_web_sm model to detect person names in free-form text. Detected names are replaced with a generic [full_name] token.

The output includes the masked text and a list of masked entities with their types and positions.

2.2 Email Classification

We implemented an email classifier that maps incoming support messages into categories Before training, each email is PII-masked to ensure the classifier doesn't rely on sensitive data.

3. Model Selection and Training

Input data was preprocessed by masking all emails using the mask_pii function. The model used was a scikit-learn pipeline:

- TfidfVectorizer: To convert text into numerical features
- RandomForestClassifier: For classification

This model was chosen for its simplicity, robustness, and ease of deployment without requiring a GPU.

4. Deployment

The API was built using FastAPI and deployed to Hugging Face Spaces using a Docker container. A Dockerfile were included to ensure smooth deployment. The endpoint is accessible with live documentation via Swagger UI.

5. Challenges & Solutions

- Masking accuracy: Combined regex and spaCy NER to cover most cases.
- Small dataset: Used masked data to reduce overfitting.
- Model loading in Docker: spaCy model was explicitly downloaded during Docker build.
- Memory/latency on Hugging Face: Used lightweight Random Forest model and slim Python image.
- Transparency: API responses include both original and masked email content.

6. Conclusion

This project successfully combines NLP and ML to automate support email classification while protecting user privacy. The FastAPI service is production-ready and deployable via Hugging Face Spaces.

Key features:
- Hybrid PII masking
- Lightweight, interpretable ML model
- Full Docker support
- API docs and testable endpoint