Report on PII Masking and Email Classification

1. Introduction

In the digital age, emails are a primary form of communication for businesses, organizations, and individuals. However, emails often contain **Personally Identifiable Information (PII)** such as full names, email addresses, phone numbers, etc. To ensure **data privacy** and compliance with regulations (e.g., GDPR), it is essential to mask or obfuscate such sensitive information before sharing or processing emails.

Additionally, businesses often receive a large volume of emails, making it necessary to classify emails into relevant categories, such as "Request," "Complaint," "Spam," etc. Automating the classification of emails can enhance productivity and streamline the handling of various requests.

This report outlines the approach and implementation of an **Email Classification and PII Masking system**, focusing on the steps taken to identify and mask PII data, classify emails into predefined categories, and the challenges faced during the process.

2. Approach Taken for PII Masking and Classification

2.1 PII Masking

The goal of PII masking is to replace identifiable information (e.g., full names, email addresses, phone numbers) with a generic tag, such as [full_name], [email], etc. This ensures that the original sensitive information is not exposed.

Steps Involved in PII Masking:

1. **Entity Recognition:**
   Using **regular expressions (regex)**, we detect common PII entities in the text such as full names, email addresses, phone numbers, dates of birth, etc. The regex patterns are designed to identify these entities with high precision.
2. **Tagging the Entities:**
   Once an entity is detected, it is replaced with a generic tag (e.g., [full_name], [email]). The regex ensures that only the PII data is replaced, while the rest of the text remains intact.
3. **Position Mapping:**
   To maintain the integrity of the original text and enable **demasking** (restoring the original text), we track the start and end positions of the masked entities. These positions help in reversing the masking process.

2.2 Email Classification

Email classification helps in categorizing emails based on their content, which makes it easier to prioritize and manage them.

Steps Involved in Email Classification:

1. **Model Selection:**
   We used a **machine learning model** for email classification. A **Random Forest Classifier** was chosen for its ability to handle large datasets and perform well with text-based features.
2. **Text Feature Extraction:**
   The email content is first converted into numerical data using **TF-IDF (Term Frequency-Inverse Document Frequency)**. TF-IDF is a technique that reflects the importance of words in a document relative to a collection of documents (or corpus).

3. **Model Training:**
   The classifier is trained on labeled email data, where each email is tagged with categories like "Request," "Complaint," etc. The model learns the relationships between email content and categories, and after training, it can classify new emails automatically.

4. **Prediction:**
   When a new email arrives, the classification model predicts the appropriate category based on the email's content.

3. Model Selection and Training Details

3.1 Model for Email Classification

The **Random Forest Classifier** was selected as the model for email classification due to its ability to manage large datasets, its robustness to overfitting, and its suitability for feature-rich datasets like emails. The training process involved the following:

1. **Data Preprocessing:**
   Emails were preprocessed by converting text into lower case, removing stop words, and applying lemmatization to reduce words to their base form.

2. **Feature Extraction:**
   The **TF-IDF vectorizer** was applied to convert the raw email text into numerical features that can be fed into the model.

3. **Model Evaluation:**
   Cross-validation was performed to assess the model's accuracy and reliability.

3.2 Model for PII Masking

For PII masking, we did not use machine learning models but relied on **regular expressions** (regex). Regular expressions provide a precise and efficient way to identify various types of PII data, such as:

- Full names (e.g., "John Smith")
- Email addresses (e.g., "john.smith@example.com")
- Phone numbers (e.g., "1234567890")

4. Challenges Faced and Solutions Implemented

4.1 Challenge: Incorrect Position Indexing After Replacements

When masking PII entities in the text, we encountered issues with incorrect positions after replacing entities. This was due to the shifting of indices as text was modified.

Solution:

- We introduced an offset mechanism to track how much the indices shift with each replacement. This ensured that the positions of subsequent entities were calculated correctly.

4.2 Challenge: Handling Punctuation in PII Entities

Some of the PII entities (e.g., full names or email addresses) contained punctuation, such as commas, periods, or question marks, which could interfere with regex matching and position tracking.

Solution:

- We stripped trailing punctuation from matched entities to ensure that the regex correctly identified the start and end of the entity, preventing unnecessary changes in the indices.

4.3 Challenge: Insufficient Labeled Data for Email Classification

A challenge in training the email classification model was the lack of a sufficiently large labeled dataset for all possible email categories.

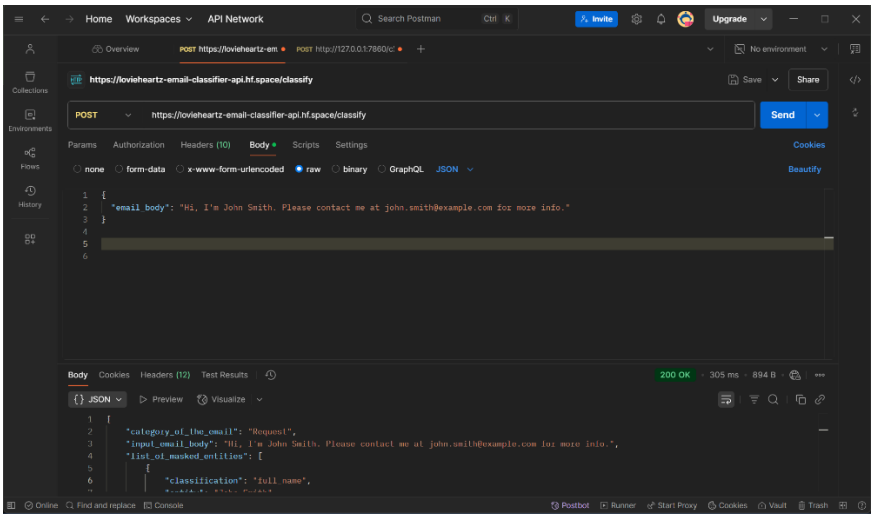
Solution:

- To overcome this, we fine-tuned the model using a **TF-IDF vectorizer** and performed **cross-validation** to optimize performance despite the small dataset. The model was able to generalize and predict email categories accurately.

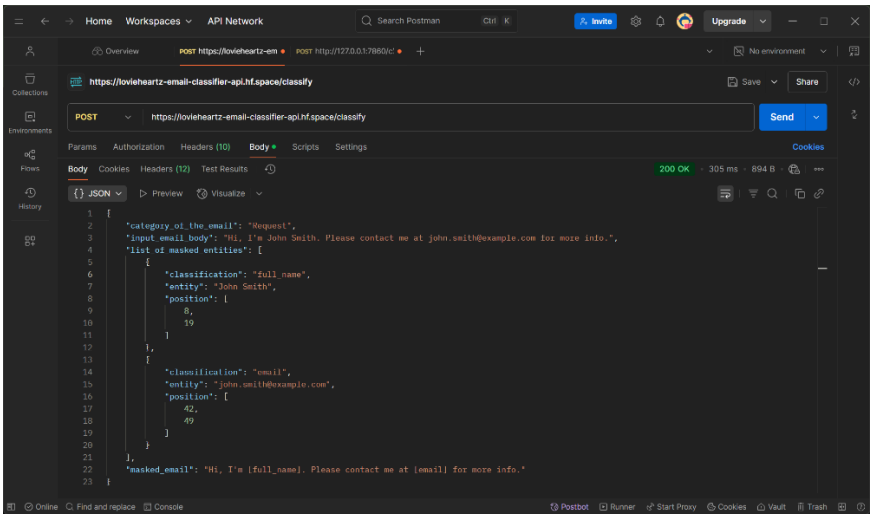
5. Final Output

5.1 API Endpoint

Input on Hugging Face URL:



Output on Hugging Face URL:



5.2 Final Output Summary:

- **Description:** Accepts an email body, masks any detected PII entities, and classifies the email into a relevant category.
- A Flask-based API that exposes endpoints for both masking and classification, making the solution easy to integrate into other systems or applications.