E-Mail Classifier

Objective

Design and implement a simple email phishing classifier.

creating a synthetic email dataset, followed by building a model to classify emails as phishing or legitimate.

Finally, create a demonstration of classifier using Hugging Face or Gradio.

Synthetic data generation

Fields:

1) Sender Email
2) Subject of Email
3) Message body of Email

Requirement - 1

1. Synthetic Email Dataset Creation:

Utilize free tools and resources to generate a synthetic dataset of email content.

Your dataset should include features relevant to phishing detection, such as the email subject, sender information, body text, and any URLs contained within the email.

Ensure your dataset has a balanced distribution of phishing and legitimate emails.

Document the process and tools used for dataset generation, including any assumptions made or limitations encountered.

Tools and Resources:

Utilized a combination of open-source libraries and online resources to generate synthetic email data. For email content generation, leveraged the mimesis library to create realistic sender email id, heuristics to generate email subjects, and body text. Additionally, used open ai API to generate email id, subject and message body. Also made use of existing real dataset from hugging face.

Other Alternative options to explore:

1) Faker library
2) https://github.com/makcedward/nlpaug
3) Tried (https://docs.sdv.dev/sdv) - no positive outcome to create data
4) Try with more heuristic based templates

Challenges

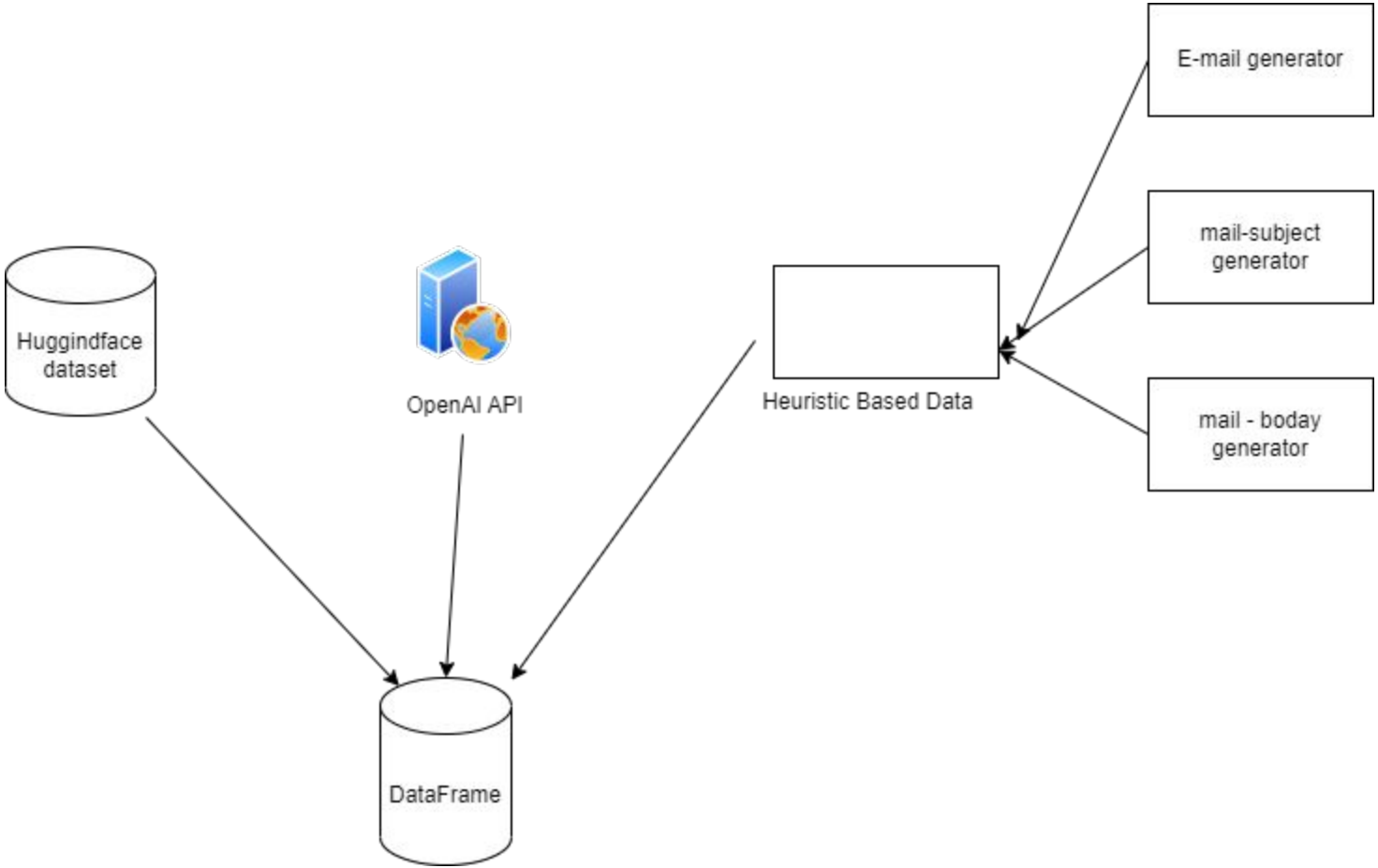
1) Creation of legitimate e-mail using heuristics
2) LLM API call limitation
3) Some of spam and legitimate e-mail will be difficult to figure out by using subject and email body

Data distribution

To ensure a balanced distribution of phishing and legitimate emails, we applied stratified dataset creation. By controlling the proportion of phishing and legitimate emails at the generation stage, we aimed to mitigate biases in the dataset and enhance the model's generalizability.

Assumptions and Limitations:

- Assumptions: Assumed that the synthesized email content would be representative of real-world email data, allowing our model to learn relevant patterns and features for phishing detection.
- Limitations: Lack of enough variability in sender information and email subjects. To address this, combined synthesized content with Huggingface dataset.


E-mail generator
mail-subject generator
mail - boday generator
Heuristic Based Data
Huggindface dataset
OpenAI API
DataFrame

Requirement - 2

Phishing Classifier Development:

Preprocess and analyze the synthetic dataset to prepare it for model training. Highlight any preprocessing steps such as text cleaning, tokenization, or feature extraction.

Build a simple machine learning model capable of classifying emails into phishing or legitimate categories. You are encouraged to explore different models and select one based on its simplicity, performance, and explainability.

Evaluate the model's performance using appropriate metrics (e.g., accuracy, precision, recall, F1 score). Discuss any potential improvements or optimizations that could enhance the model's effectiveness.

Phishing Classifier Development

Preprocessing and Analysis:

Text Cleaning: Performed standard text preprocessing steps, including lowercasing, removing special characters, and lemmatization, to ensure consistency in the textual data.

Tokenization: The email body text was tokenized using NLTK's word_tokenize function, splitting the text into individual words or tokens for further analysis.

Feature Extraction: Used TF-IDF (Term Frequency-Inverse Document Frequency) vectorization to convert the text data into numerical features, capturing the importance of words in each email relative to the entire dataset

Model Building and Evaluation:

Model Selection: Experimented with various machine learning models, including logistic regression, random forest, and support vector machines (SVM), to develop a phishing classifier. Based on performance, simplicity selected the logistic regression model .

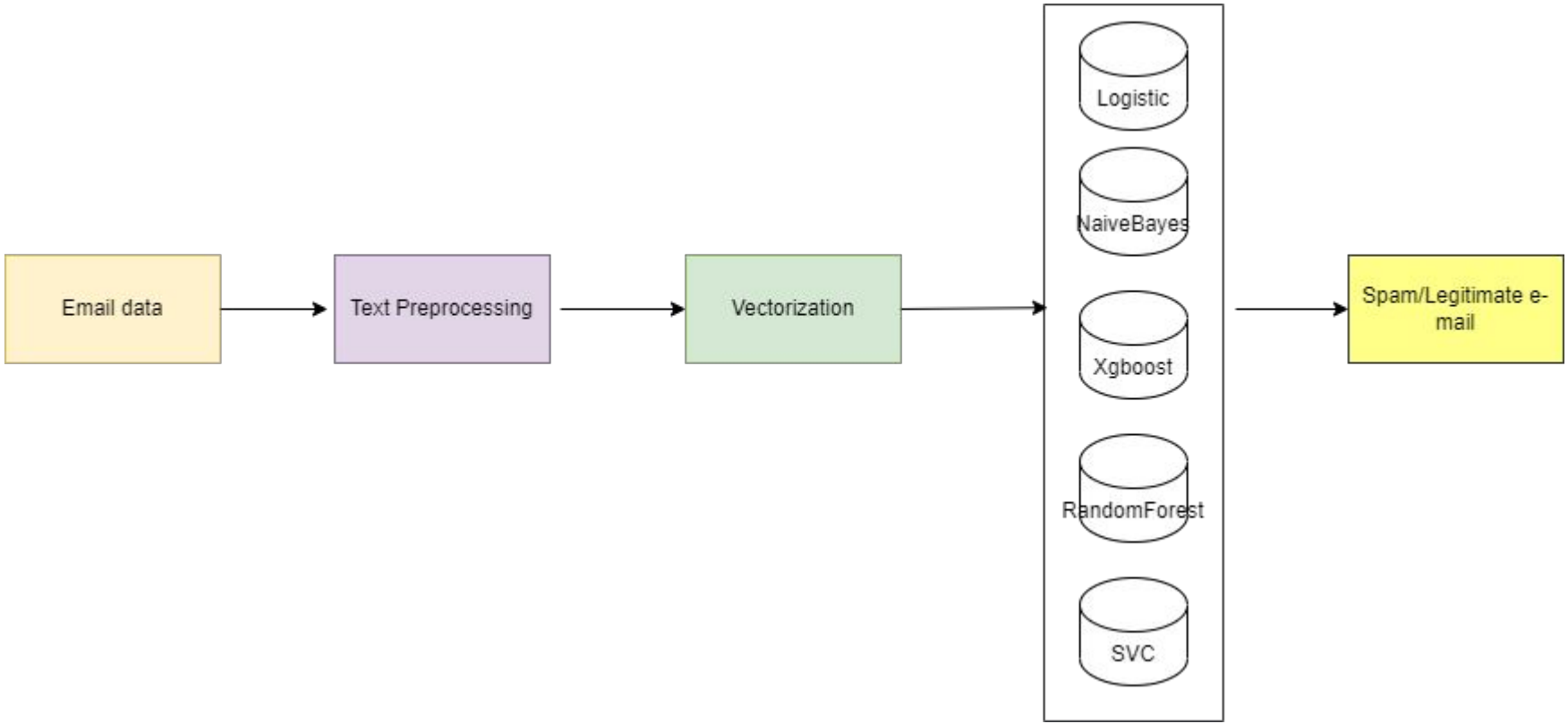
Performance Metrics: The model's performance was evaluated using standard classification metrics such as accuracy, precision, recall, and F1 score, computed on a held-out validation set.

To explore

- feature engineering - add additional signals to the model.
- Experiment with ensemble methods or deep learning architectures for improved performance.

Improvements & optimization

1) Hyperparam tuning instead of default params of model, tfidf vectorizer.
2) Gather more hard samples to check the performance of classifier -


Email data
Text Preprocessing
Vectorization
Logistic
NaiveBayes
Xgboost
RandomForest
SVC
Spam/Legitimate e-mail

Requirement - 3

Develop a simple web application using Hugging Face Spaces or Gradio that allows

users to input email content and receive a classification result indicating whether

the email is phishing or legitimate.

The application should provide a user-friendly interface and display the

classification result clearly, along with any confidence scores or relevant

information supporting the decision.

Include a brief user guide or documentation within the application to assist users in

navigating and understanding the output.

Deployment

1) A lightweight flask deployment.
2) Gradio app with UI to take user input and generate output

Conclusion

1) **Current synthetic data generation can be improved, adding more samples can help with cases where model is confused (49%,51%) for the classes.**
2) **Current training didn't add the sender e-mail into account - Most of the phishing e-mail may look legitimate(sub,body) - could be identified only by looking into the e-mail address and link url present in the mail.**
3) **Hyperparam tuning to get the best model.**
4) **Inference result - ensemble of different models .**
5) **Other strategy to train - Instead of ML models, can try with BERT or decoder only model by attaching classification head and fine-tuning them(compute,time cost increase, decrease in explainability)**