E-Mail thread classifier

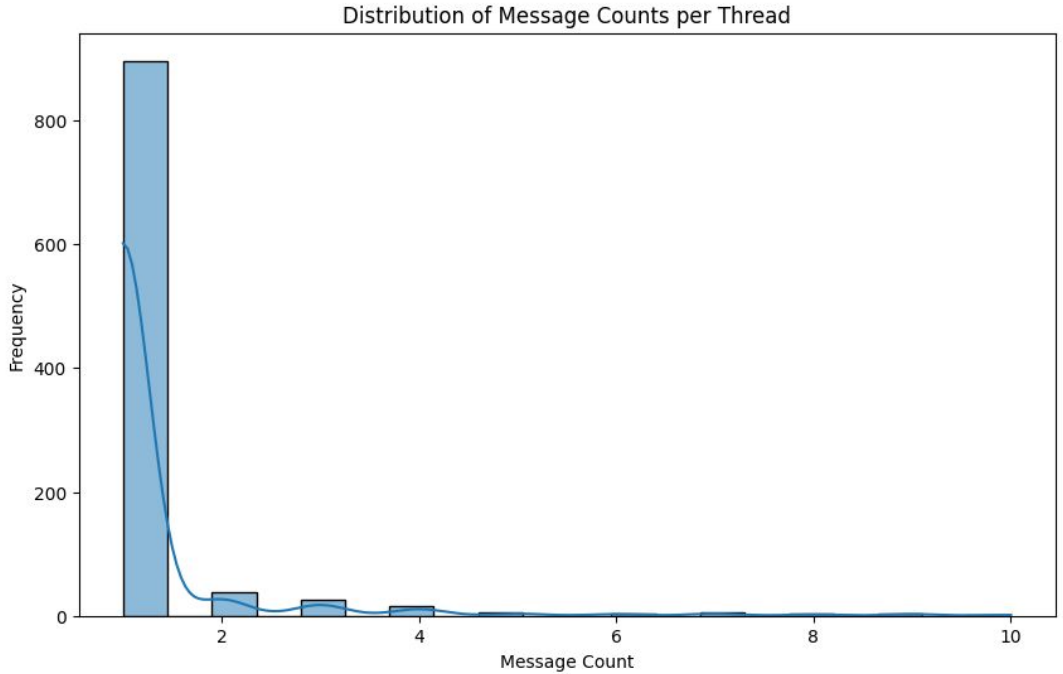
Objective

The goal of assignment is to demonstrate your ability to derive meaningful insights and build machine learning models from a real-world real estate data set.

1. **Exploratory analysis on inbox**

Exploratory analysis on this inbox

Observation regarding the number of emails per thread:



Exploratory analysis on this inbox

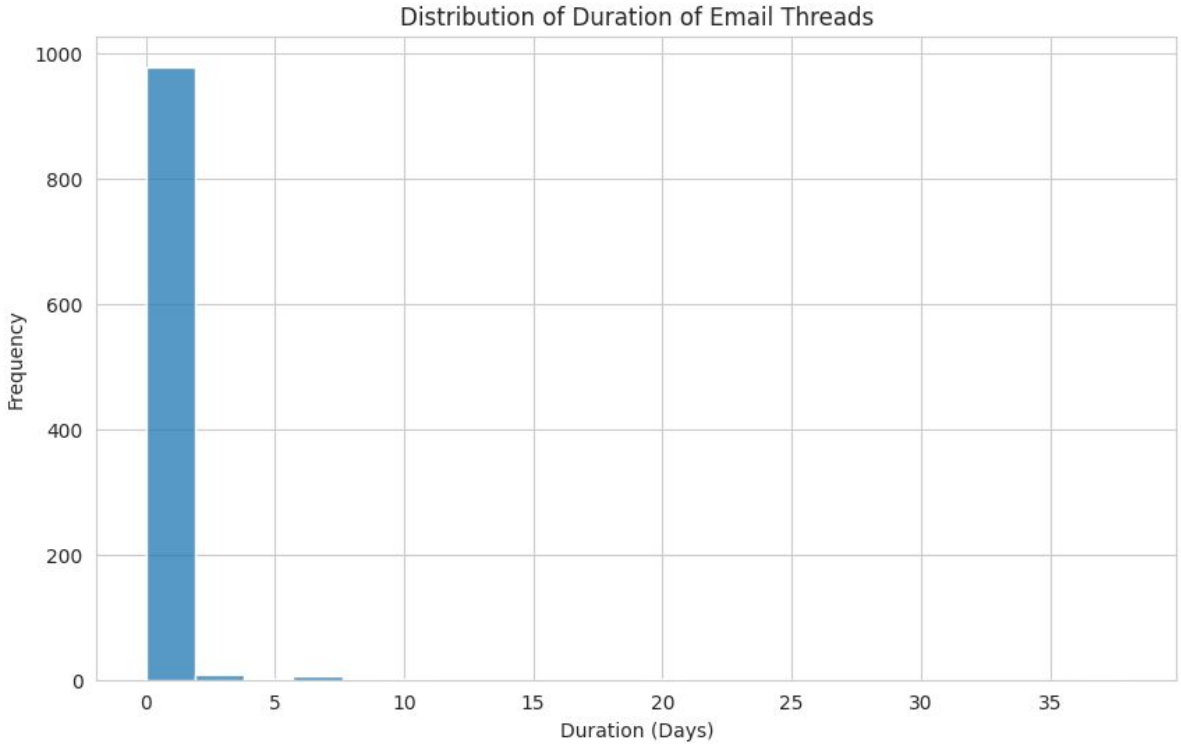
Observation regarding the number of emails per thread:

	thread_id	message_count
0	18eaa95659e46ced	10
1	18db381d2257822d	10
2	18ee3711066442c6	9
3	18da2d3871e15e8b	9
4	18ecf4dbc862dbed	9
5	18d574676c555b09	9
6	18d47b90f85f66b7	8
7	18ecde24026659c1	8
8	18ecf27cc3e6f4a9	8
9	18e10ac043947d39	7
10	18eaec866b29d9be	7

Based on the observation, its noticed that ~17 threads contain messages > 5. Most of the threads have single message

Exploratory analysis on this inbox

observation regarding the duration of the email thread?



Exploratory analysis on this inbox

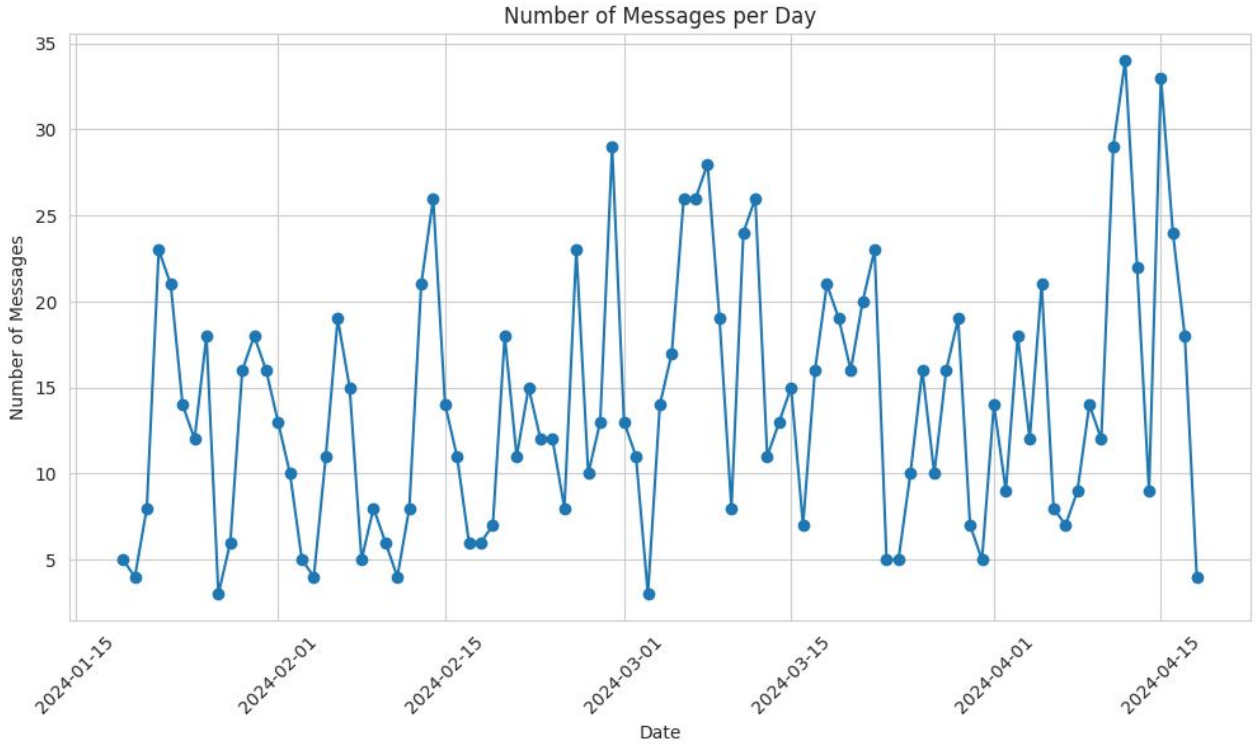
observation regarding the duration of the email thread?

```
count                          1000
mean         0 days 04:52:26.873000
std       1 days 16:54:18.920229757
min                  0 days 00:00:00
25%                  0 days 00:00:00
50%                  0 days 00:00:00
75%                  0 days 00:00:00
max                 38 days 19:57:39
Name: date, dtype: object
```

It is observed that mean time to close a thread is ~4hrs and max time to close a thread is 38 days. As most of the thread contains just a single e-mail the close time will be zero days

Exploratory analysis on this inbox

Other observations on data - Messages per day



Exploratory analysis on this inbox

Other observations on data - Sender stats

	sender	num_non_spam	num_spam	total_count
189	Redfin <listings@redfin.com>	0	161	161
139	Larry Schneider <larry@jasonmitchellgroup.com>	120	3	123
31	"realtor.com" <consumer@e.mail.realtor.com>	0	119	119
255	connectMLS <no-reply@connectmls.com>	0	47	47
140	Larry Schneider <noreply@skyslope.com>	38	4	42
...	...	...	...	...
75	Chicago Title <noreply@inhere.com>	1	0	1
163	Matt at Showami <mattkuchar@showami.com>	0	1	1
76	Chris Herb <cherb@jasonmitchellgroup.com>	1	0	1

Exploratory analysis on this inbox

Other observations on data - Sender stats

	sender	num_non_spam	num_spam	total_count
189	Redfin <listings@redfin.com>	0	161	161
139	Larry Schneider <larry@jasonmitchellgroup.com>	120	3	123
31	"realtor.com" <consumer@e.mail.realtor.com>	0	119	119
255	connectMLS <no-reply@connectmls.com>	0	47	47
140	Larry Schneider <noreply@skyslope.com>	38	4	42
...	...	...	...	...
75	Chicago Title <noreply@inhere.com>	1	0	1
163	Matt at Showami <mattkuchar@showami.com>	0	1	1
76	Chris Herb <cherb@jasonmitchellgroup.com>	1	0	1

Exploratory analysis on this inbox

Please refer jupyter notebook for other EDA Details

2. Deeper analysis on inbox and assumptions

Deeper Analysis of inbox

Emails that are referring to a specific real estate transaction

Assumptions: As its not mentioned/described what defines a real estate transaction, It is assumed that if the email have any of the following keywords it is related to transaction

```
transaction_keywords = ["property", "buy", "sell", "rent", "listing", "house",
"apartment",  "transaction","escrow", "attorney", "inspection",
"appraisal","mortgage", "loan", "financing", "down payment","purchase",
"offer", "contract", "closing"]
```

Out of ~1.2K messages 1077 messages have one of the above keywords stating the mail is related to transaction. Most of the spam mail are not random spam and is closely related to real estate spams

Deeper Analysis of inbox

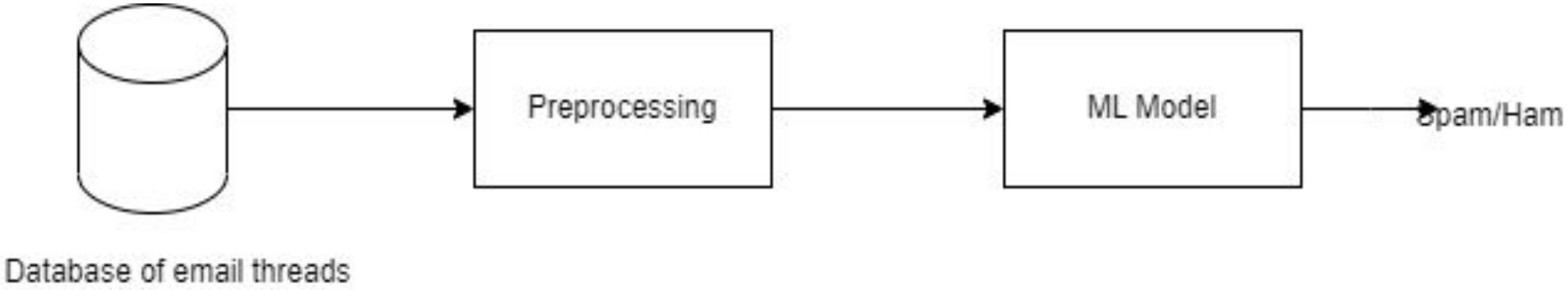
Top 3-5 topics

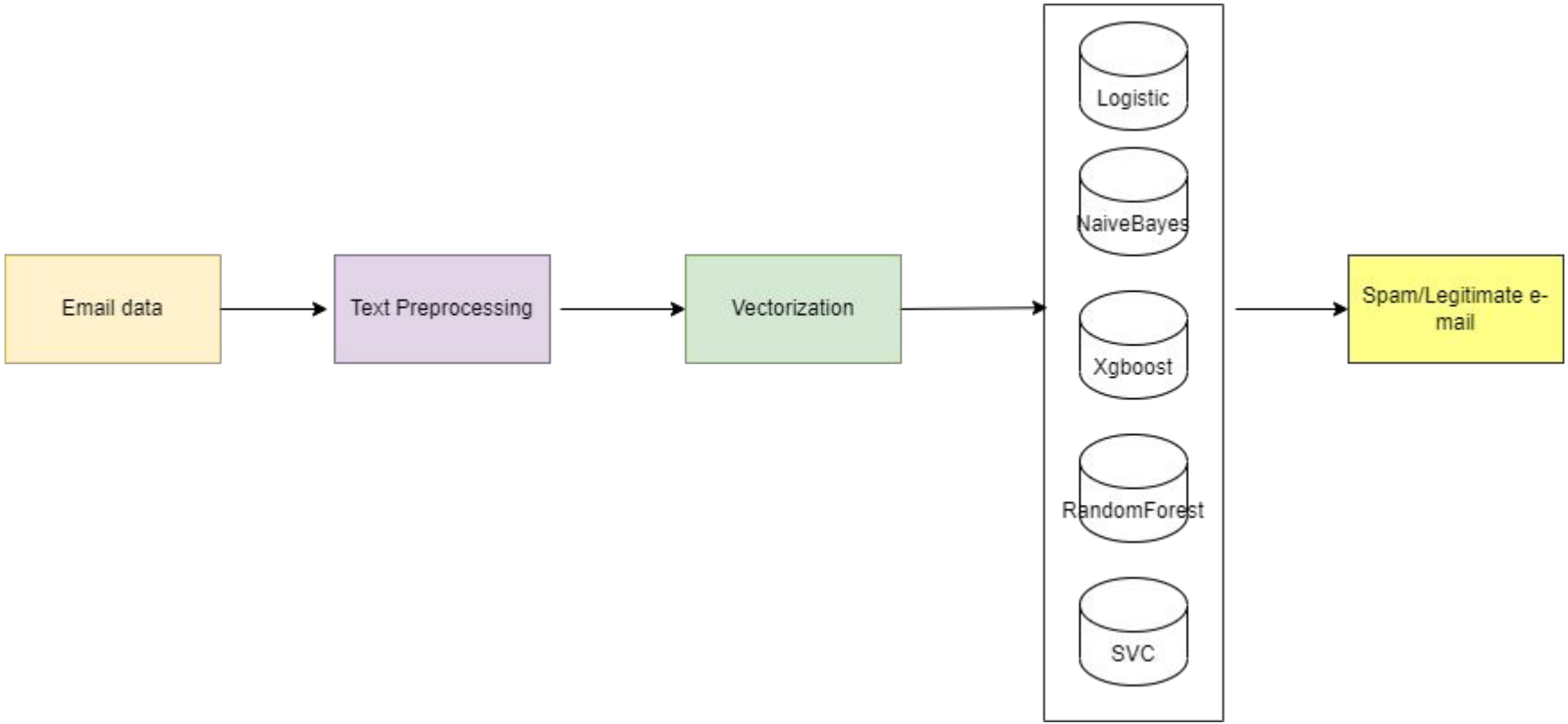
```
LDA Topics:
Topic 1: 0.058*"redfin" + 0.045*"recommend" + 0.038*"price" + 0.036*
Topic 2: 0.112*"sqft" + 0.081*"sale" + 0.080*"bath" + 0.079*"bed" +
Topic 3: 0.014*"document" + 0.014*"larri" + 0.013*"thank" + 0.013*"i
Topic 4: 0.013*"client" + 0.011*"real" + 0.011*"estat" + 0.009*"new"
Topic 5: 0.015*"wire" + 0.012*"com" + 0.012*"inform" + 0.011*"loan"
```

Please refer jupyter notebook for topic analysis

3. Spam detector on the inbox data

High Level system architecture




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

Spam 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

Spam Classifier Development

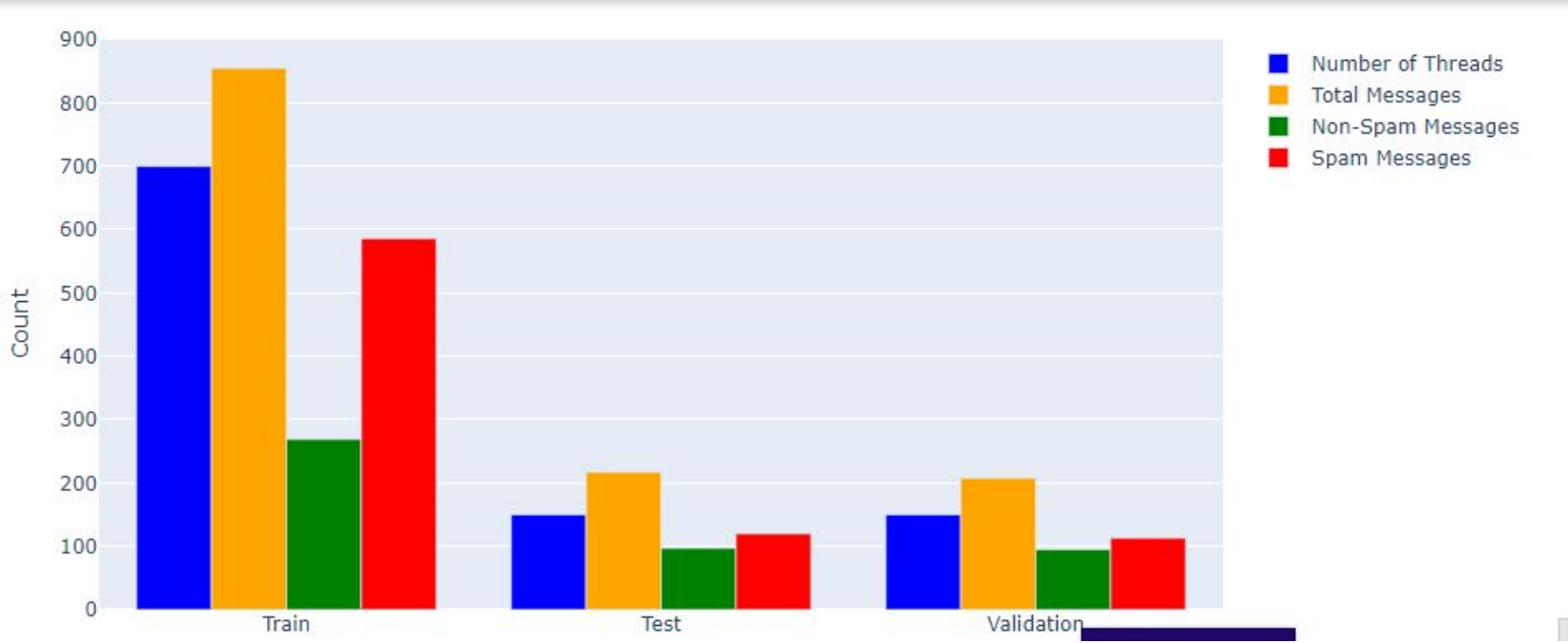
How is the data split created?

As the row in the dataset represents a single message, we can’t split the data directly by rows as it may result in data leakage.

We split the data by thread_id and did a stratified sampling across train,test,val split.

Spam Classifier Development

How is the data split created?



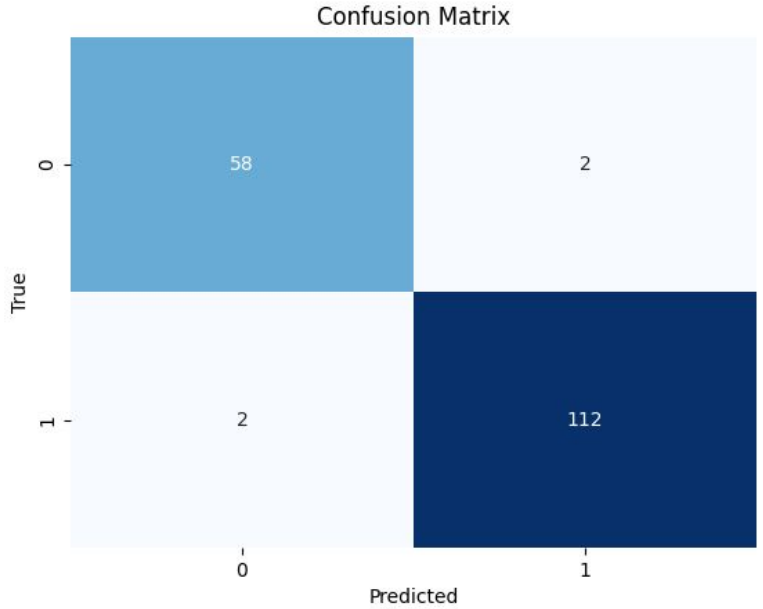
What features are for Spam detector? What's your choice of model and why?

- **For the simplicity, we have taken plain_text_body as the feature for spam detector and also text_body contains more information to understand if a mail is spam or ham.**

- **Regarding the choice of model, tried various ML models and chose Logistic Regression. Why? All the model results are close to similar accuracy and as logistic regression model is easier to interpret, quicker to inference and prototype it was default choice**

Performance of model?

- 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.



How can the performance be improved?

- As the test size is small, the current performance may not be the actual performance of the model.
- Using current model - We can try hyperparam tuning to improve the performance
- Balance the dataset, create synthetic data using LLMS for genuine real state email thread
- Improve the data - Currently we didn’t do feature engineering. We can do and increase the num of features.

Other Alternatives

- Try using open(BERT)/closed source LLM - Trade off is cost Vs performance. Latency in prediction. Context length. Some of the email will have larger context length if we pass the raw text which can contain HTML,CSS.

- Create synthetic data for ham emails as our goal is to have higher recall for HAM email.

- By using additional features like subject, sender e-mail can also help. Creating new features like num_hyperlinks, num_special_chars, domain_of_sender, len_of_subject, len_of_body

Other Alternatives

- Better preprocessing of the subject - NER tagging for filtering more noise in body, instead of the body itself - create summary using LLM and use it as feature as the LLM can do processing of removing noise and create a crisp summary - Trade off is there should a way to eval if the summary is correct, extra call to LLM to create summary

- Try using zero-shot classification - May fail if we give the raw body text as models will have context length limitations

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 -

Conclusion

1) **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.**
2) **Hyperparam tuning to get the best model.**
3) **Inference result - ensemble of different models .**
4) **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)**

Any other questions/suggestions - Feel free to reach out on vivek.mail2022@gmail.com