



# COS720 Semester Project - Phisherman

2025

---

Mbofho Mamatsharaga

18045881

University of Pretoria

## Research Overview

### Dataset Preparation and Model Development

#### CEAS 08 dataset:

Are there any missing values?

What is the balance between legit and non legit?

Cleaning done so far?

Split between Train set and test set?

Results

Confusion Matrix

#### Enron Dataset:

Are there any missing values?

Balance between legit and phishing?

Cleaning done so far?

Split between train and test set?

Results:

Confusion Matrix

#### Ling Dataset:

#### Nazario Dataset:

#### Nigerian Fraud dataset:

#### Phishing Emails dataset:

Are there any missing values?

Balance between Legit and Phishing emails?

Cleaning done so far?

Split between Train and Test sets?

Results:

1. Logistic Regression

Confusion Matrix:

2. Naive Bayes

Confusion Matrix:

3. SVM

Confusion Matrix

4. Random Forest

Confusion Matrix:

#### Spam Assassin Dataset

Are there any missing values?

What is the balance between legit and phishing emails?

[Cleaning done so far?](#)

[Split between train and test set?](#)

[Results:](#)

[Confusion Matrix:](#)

[Possible Limitations:](#)

[Model Selection Process:](#)

[Similar Tools](#)

[Graphus:](#)

[Email Veritas Security Technologies Phishing Detector for Outlook:](#)

[Visua:](#)

[UML Diagrams:](#)

[How it should work.](#)

[How Phisherman works:](#)

[Bibliography](#)

## Research Overview

As the world goes digital, the world's digital advantages have come with their fair share of disadvantages. Phishing emails being one of them. Phishing emails have been a huge problem for companies and individuals worldwide. It has spread at an exponential rate since the world went digital, and with that, attackers have become smarter, finding ways to bypass traditional spam filters using social engineering. With this project I am exploring the idea of using AI to combat phishing email attacks. I used a set of Kaggle datasets in my preparation and landed on using the CEAS\_08 dataset as my final databaset as it made the most sense to me so far. Using term frequency vectorization to preprocess the dataset has helped me in preparation for using learning models and finally landing on Logistic regression to train the dataset and on to get the most accurate result (99% accuracy). I then developed a web based interface using python's streamlit framework as this made everything a lot easier because I was already using python to prep and train the model.

In the essence what I am trying to do here with detecting phishing using AI is taking a set of existing emails and using those as a guideline as to if an email I receive is a legitimate email or a phishing email.

How this all works is that the Machine learning/AI model detects AI by analyzing patterns, content and grammar in given emails and in order to determine if the email is phishing or not. I train machine learning models on existing datasets of emails that are labeled to be

phishing or legitimate emails. In my case I'm using datasets from Kaggle as mentioned before. With this dataset I train the model to identify patterns and cues in emails and turn them into statistics. It looks for things such as poor grammar, unusual words, domains in links provided, and if the emails are incorrectly labeled as urgent. I use NLP techniques to pre process the data. In this case for all models I used Term Frequency vectorization to convert the data it scrapes in the emails into vectors that the ML model (in my case Logistic regression) can process so that I can train the models and flag emails as accurately as possible. Because the dataset has now been vectorized, I can then also use it to calculate confidence scores for it's decisions. Theoretically, this should then allow us to keep up to date with the latest phishing techniques as the datasets keep getting updated, however as of right now that is not how my system is setup.

My solution is called Phisherman.

## Dataset Preparation and Model Development

I attempted to train all the datasets Kaggle provided for phishing detection and I came to the following conclusions:

### CEAS\_08 dataset:

```
In [1]: import pandas as pd
        df = pd.read_csv('CEAS_08.csv')
        print(df.head())
```

Are there any missing values?

Yes. I then trimmed the data by deleting any entries with null fields.

What is the balance between legit and non legit?

1 21827

0 16842

Name: label, dtype: int64

56/44

Where 1 = phishing email and 0 = legitimate email.

This means the balance is Acceptable. No need for further action is needed.

Cleaning done so far?

1. Lowercase
2. Remove extra spaces

Split between Train set and test set?

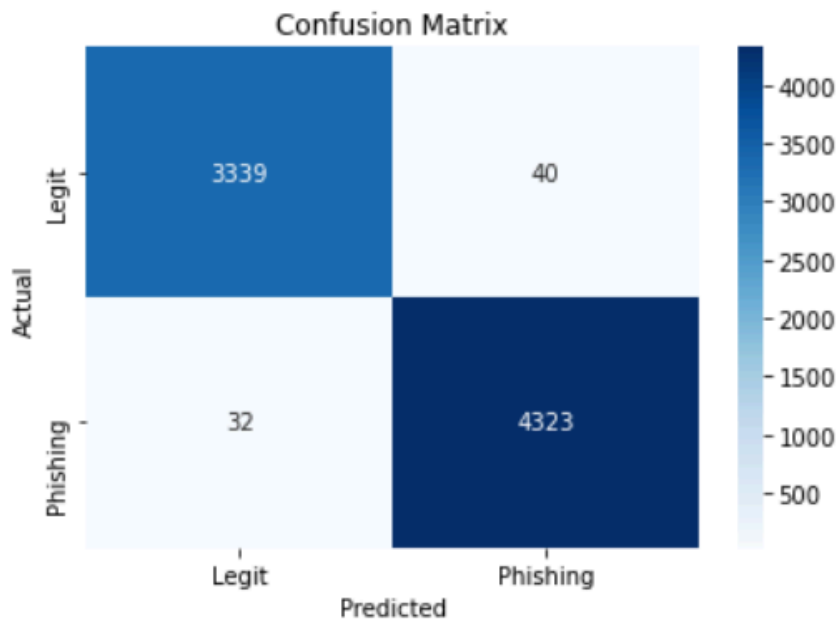
Yes. 80/20 split (Apparently standard)

Results

Accuracy: 0.9906904577191621

Classification Report:					
	precision	recall	f1-score	support	
0	0.99	0.99	0.99	3379	
1	0.99	0.99	0.99	4355	
accuracy			0.99	7734	
macro avg	0.99	0.99	0.99	7734	
weighted avg	0.99	0.99	0.99	7734	

Confusion Matrix



## Enron Dataset:

```
In [1]: import pandas as pd
df = pd.read_csv('Enron.csv')
print(df.head())
```

Are there any missing values?

Yes. I then trimmed the data by deleting any entries with null fields.

Balance between legit and phishing?

```
In [5]: print(df['label'].value_counts())
0      15791
1      13778
Name: label, dtype: int64
```

53/47

Where 1 = phishing email and 0 = legitimate email.

This means the balance is Acceptable. No further action is needed.

Cleaning done so far?

1. Lowercase
2. Remove extra spaces

Split between train and test set?

Yes. 80/20 split (Apparently standard)

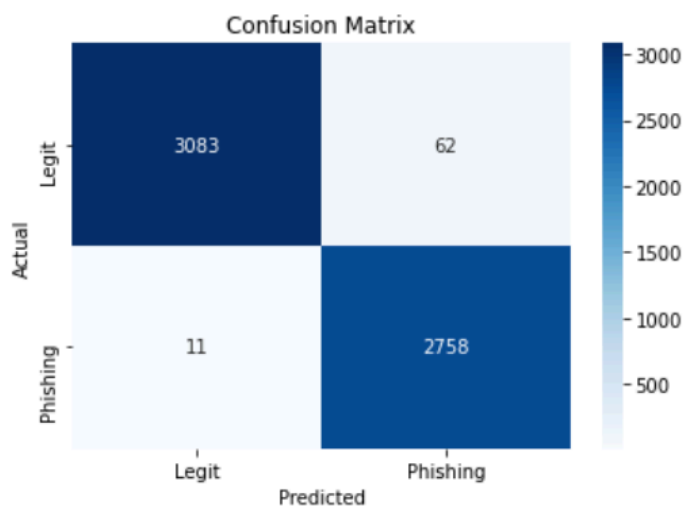
Results:

Accuracy: 0.9876564085221509

#### Classification Report:

	precision	recall	f1-score	support
0	1.00	0.98	0.99	3145
1	0.98	1.00	0.99	2769
accuracy			0.99	5914
macro avg	0.99	0.99	0.99	5914
weighted avg	0.99	0.99	0.99	5914

#### Confusion Matrix



#### Ling Dataset:

Invalid for classification. Balance is 86/14. This is problematic because this means that the model would favour one label over the other if it is trained on this dataset.

## Nazario Dataset:

Invalid for classification. Balance is 100/0. This is problematic because this means that the model would favour one label over the other if it is trained on this dataset.

## Nigerian Fraud dataset:

Invalid for classification. Balance is 100/0. This is problematic because this means that the model would favour one label over the other if it is trained on this dataset.

## Phishing Emails dataset:

```
In [1]: import pandas as pd
df = pd.read_csv('phishing_email.csv')
print(df.head())
```

Are there any missing values?

```
In [2]: print(df.isnull().sum())
```

```
text_combined    0
label            0
dtype: int64
```

No. No need to trim

Balance between Legit and Phishing emails?

```
In [3]: print(df['label'].value_counts())
```

```
1    42891
0    39595
Name: label, dtype: int64
```

52/48

Where 1 = phishing email and 0 = legitimate email.

This means the balance is Acceptable. No further action is needed.



Cleaning done so far?

1. Lowercase
2. Remove extra spaces

Split between Train and Test sets?

Yes. 80/20 split (Apparently standard)

Results:

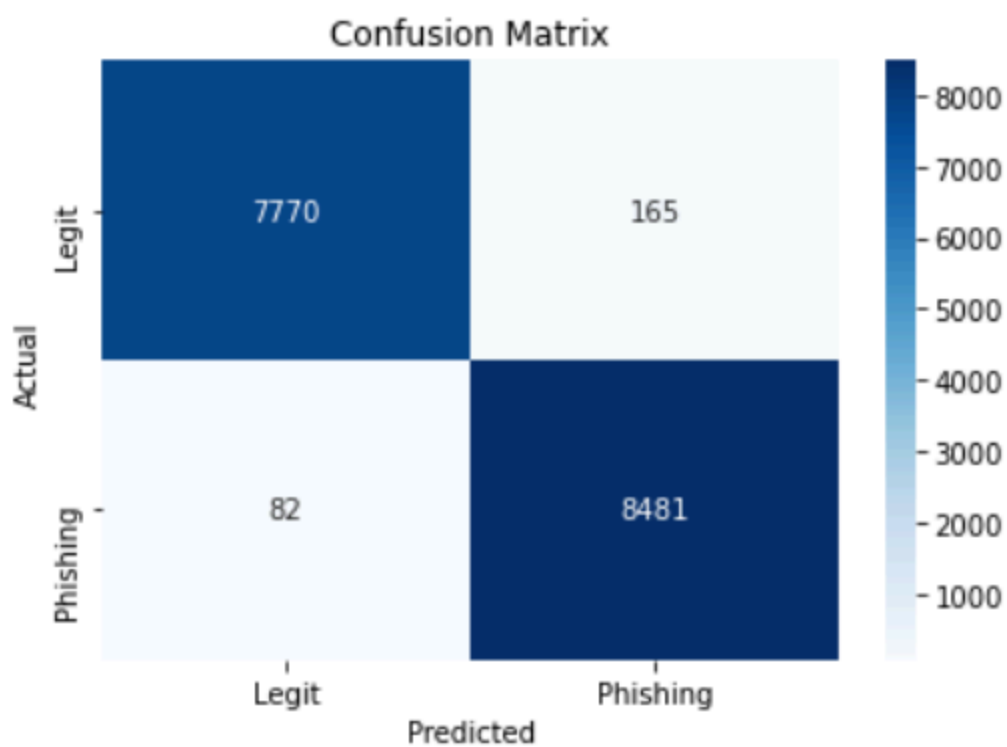
4 models were used.

1. Logistic Regression

**Accuracy: 0.9850**

	precision	recall	f1-score	support
0	0.99	0.98	0.98	7935
1	0.98	0.99	0.99	8563
accuracy			0.99	16498
macro avg	0.99	0.98	0.99	16498
weighted avg	0.99	0.99	0.99	16498

Confusion Matrix:

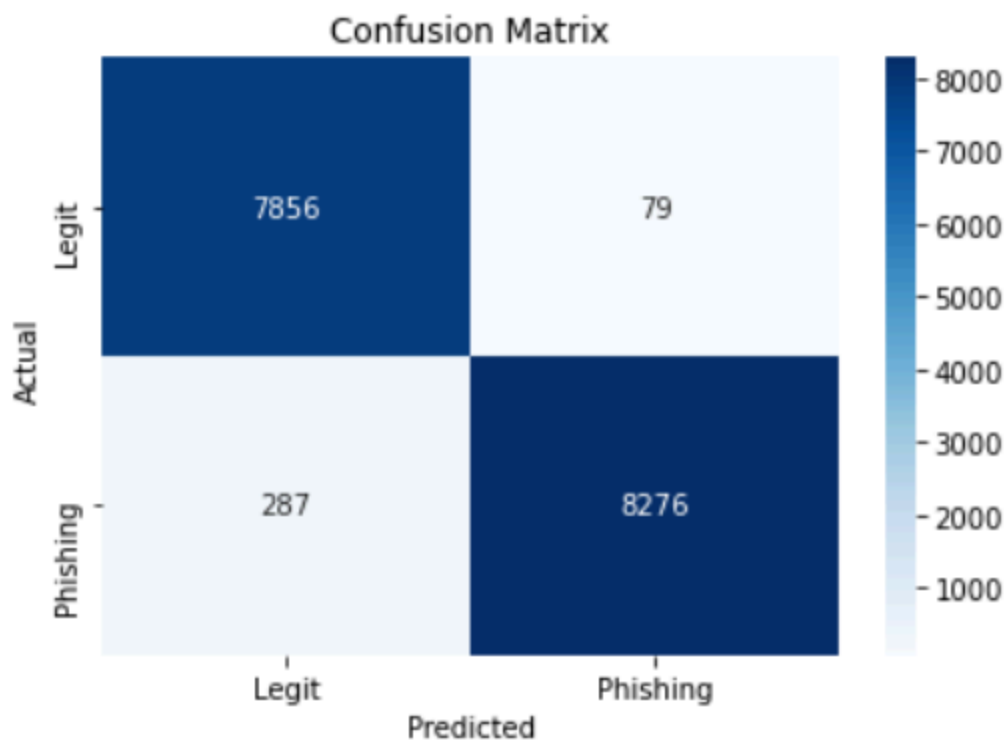


## 2. Naive Bayes

Accuracy: 0.9778

	precision	recall	f1-score	support
0	0.96	0.99	0.98	7935
1	0.99	0.97	0.98	8563
accuracy			0.98	16498
macro avg	0.98	0.98	0.98	16498
weighted avg	0.98	0.98	0.98	16498

Confusion Matrix:

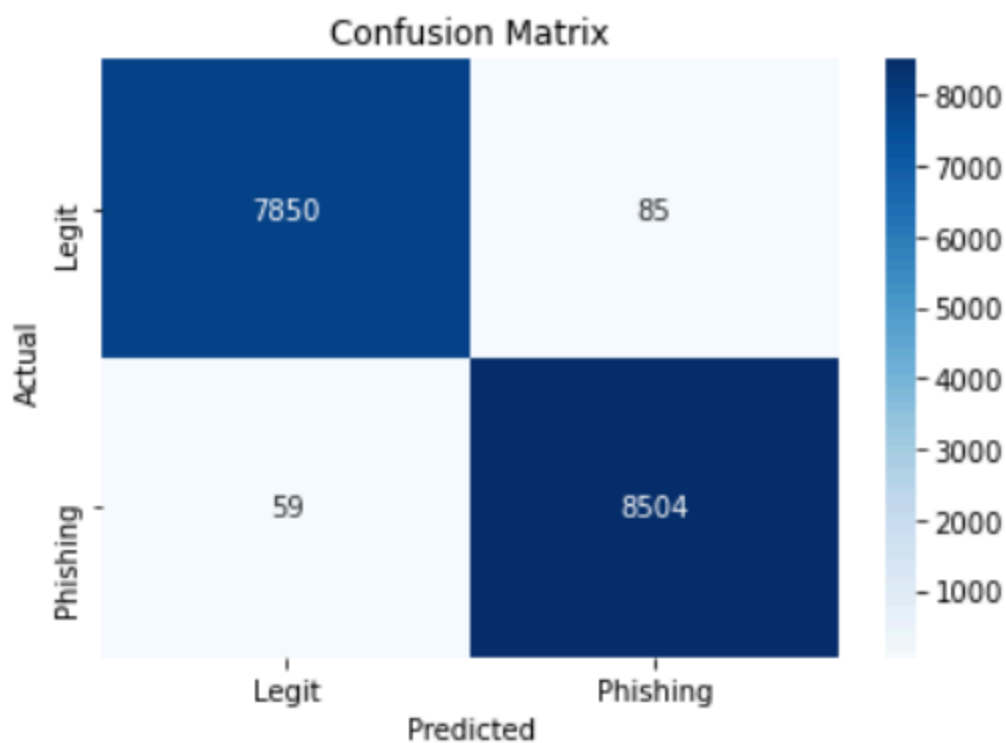


### 3. SVM

Accuracy: 0.9913

	precision	recall	f1-score	support
0	0.99	0.99	0.99	7935
1	0.99	0.99	0.99	8563
accuracy			0.99	16498
macro avg	0.99	0.99	0.99	16498
weighted avg	0.99	0.99	0.99	16498

Confusion Matrix



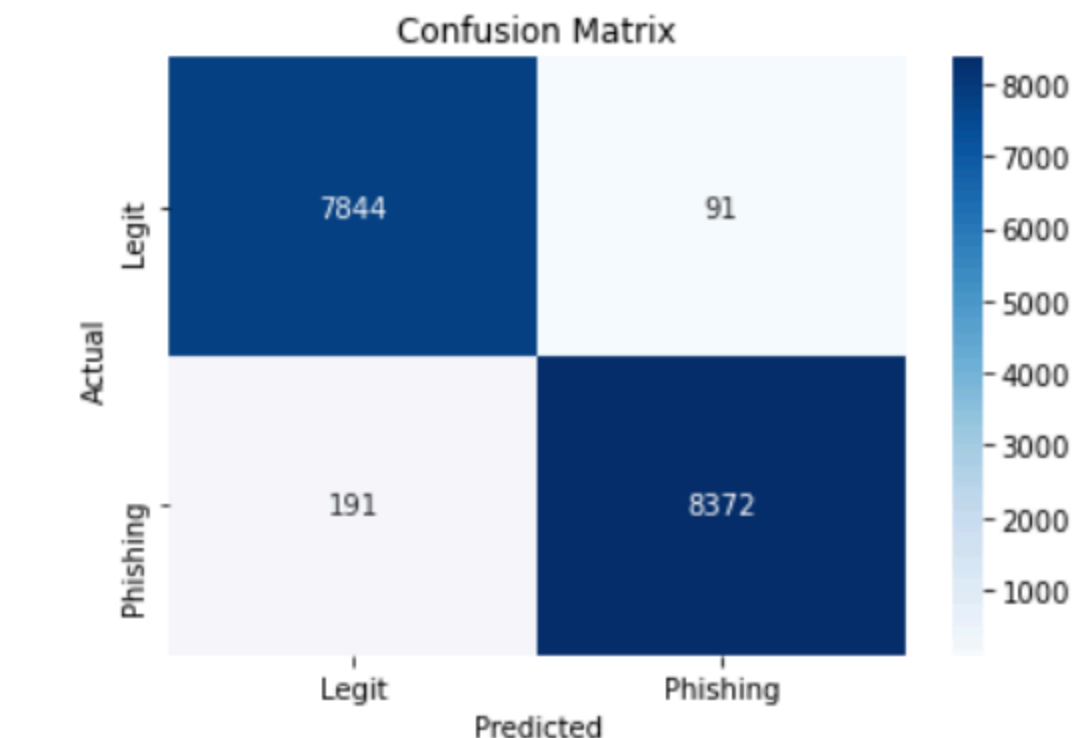
16498

4. Random Forest

Accuracy: 0.9829

	precision	recall	f1-score	support
0	0.98	0.99	0.98	7935
1	0.99	0.98	0.98	8563
accuracy			0.98	16498
macro avg	0.98	0.98	0.98	16498
weighted avg	0.98	0.98	0.98	16498

Confusion Matrix:



## Spam Assassin Dataset

```
In [1]: import pandas as pd
df = pd.read_csv('SpamAssassin.csv')
print(df.head())
```

Are there any missing values?

Yes. I then trimmed the data by deleting any entries with null fields.

What is the balance between legit and phishing emails?

```
In [5]: print(df['label'].value_counts())
0      3924
1      1658
Name: label, dtype: int64
```

70/30

Where 1 = phishing email and 0 = legitimate email.

This means the balance is not Acceptable. But I tried to balance this one as well.

```
In [11]: from sklearn.linear_model import LogisticRegression

model = LogisticRegression(class_weight='balanced')
model.fit(X_train_vec, y_train)
```

```
Out[11]: LogisticRegression(class_weight='balanced')
```

Cleaning done so far?

1. Lowercase
2. Remove extra spaces

Split between train and test set?

Yes. 80/20 split (Apparently standard)

```
In [9]: from sklearn.model_selection import train_test_split

X = df['body']
y = df['label']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Results:

Accuracy: 0.981199641897941

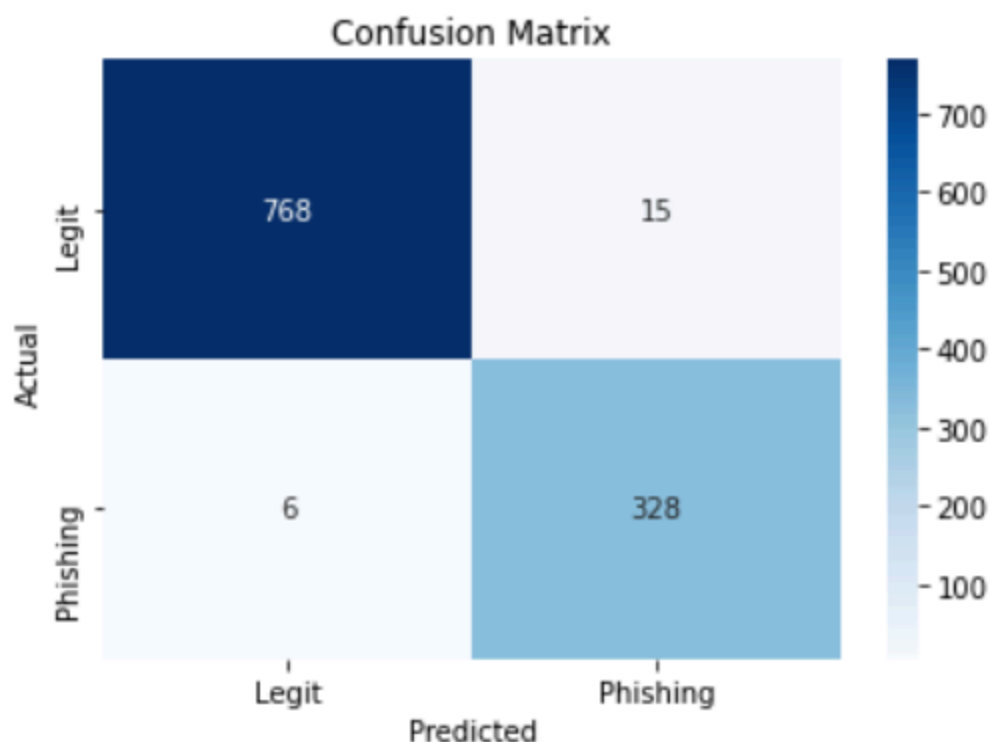
Confusion Matrix:

```
[[768 15]
 [ 6 328]]
```

Classification Report:

	precision	recall	f1-score	support
0	0.99	0.98	0.99	783
1	0.96	0.98	0.97	334
accuracy			0.98	1117
macro avg	0.97	0.98	0.98	1117
weighted avg	0.98	0.98	0.98	1117

Confusion Matrix:




## Possible Limitations:

The dataset could be a huge hamstring as far as this system goes. The balance between legitimate and phishing emails is a very important consideration when trying to train the dataset as you don't want a model that is going to have any biases. Another issue is that this system isn't able to pick up on advanced social engineering methods and grabify links as grabify links can use official domains to grab user information.

## Model Selection Process:

How I went about selecting my model was very manual. First thing was finding a dataset. I found 7 datasets that were provided via Kaggle. The datasets then had to be filtered out, after initial inspection I looked for datasets with a good balance between phishing emails and legitimate emails. The threshold I put for myself was anywhere between 60/40 and 50/50. The reason this is important is because I do not want to train a model to be biased any which way. So after this process I was left with 4 datasets that were usable, which is still 3 too many. I had to then look further into semantics, but with the remaining datasets I tried to train them on different models. 3 were trained only on Logistic Regression, and the 1 remaining one (the largest dataset, Phishing Emails Dataset) was trained on 4 models namely: Logistic Regression, Naive Bayes, SVM, and Random Forest). After this process, I looked at the different accuracies I was able to narrow it down to two datasets, namely:



Phishing Emails and CEAS\_08. I chose these two because I was able to get an accuracy score of 99% from both of them using Logistic Regression for the CEAS\_08 dataset and using the SVM model for the Phishing Emails dataset. After this I had to use other metrics to decide which dataset and which dataset and model I would use. I decided to go with the CEAS\_08 dataset because it contains more metadata and would be more useful in real world applications. Phishing Emails, as much as it's very comprehensive, only contain the email body and a label. So I ended landed on CEAS\_08 with the Logistic Regression model. This worked well for me with a relatively low number of false positives. The confusion matrix indicated 0.93% false positive cases for the CEAS\_08 dataset with Logistic Regression, which is pretty good.

## Similar Tools

### Graphus:

Graphus is a tool that uses its own algorithm called the TrustGraph algorithm, this works by evaluating each received email against many points of comparison, including historical data from the company's employees interactions and received emails. It is built on graph theory and claims to be 40% more effective in stopping malicious emails. The website breaks down the concepts of phishing emails and how their system works very well. It makes it such that even a novice to computers and the concept of AI and phishing can understand what they are paying for. It is however an enterprise level tool so there is the disadvantage that individual users can't buy their services. Here is a link to their site: [Graphus](#)

### Email Veritas Security Technologies Phishing Detector for Outlook:

EmailVeritas is another tool that can be used for Microsoft systems. This one is more of a broad user system, as you use it you can start with a 30 day free trial for up to 10 users, which means it is possible to get it on an individual basis, however it seems to use "regular" machine learning models to detect AI and flag it. I am not sure on the pricing as i didn't try it but it seems fairly honest. It doesn't have much information in the Microsoft app store page. Here is a link to it: [EmailVeritas](#)

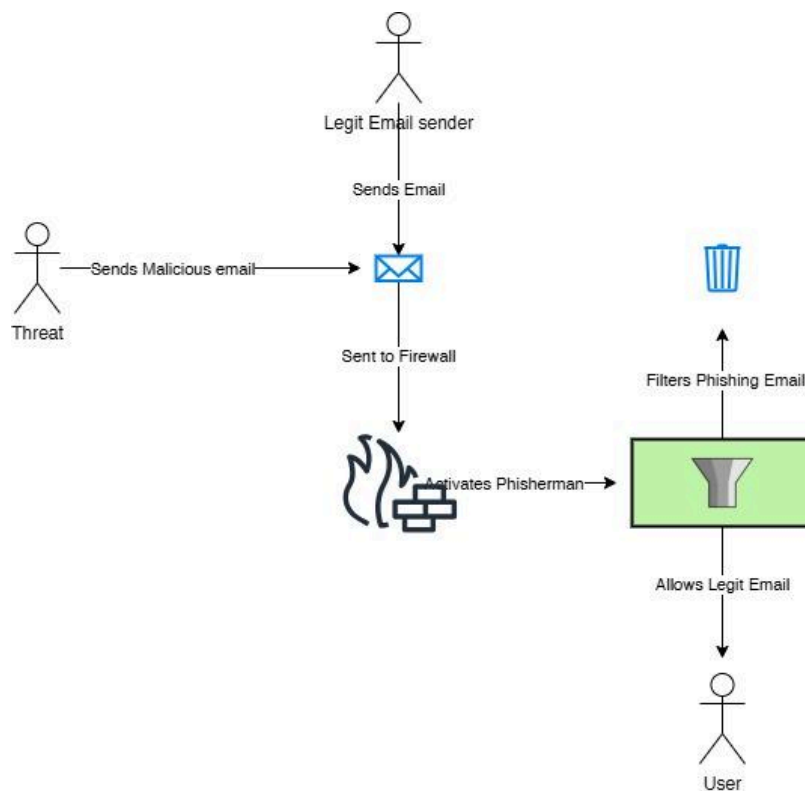
### Visua:



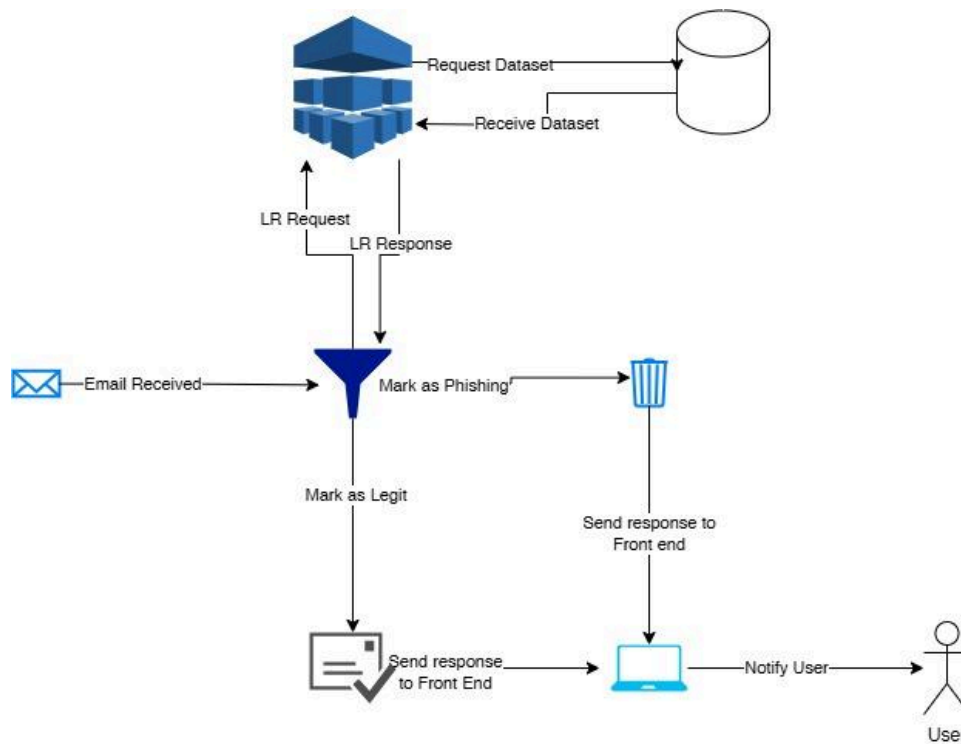
Visua can be seen as a tool similar to Graphus but instead of any backend heavy code it uses more Visual AI detection to detect phishing email by looking for visual cues, like the company logos, it does visual searches over the emails and also uses text analysis to find cues and label emails as phishing if they are. It then indicates how many threat points are raising suspicion. It seems also to be an enterprise level but I think it does have individual support too. Here is the link: [Visua](#)

## UML Diagrams:

How it should work.



How Phisherman works:



## Bibliography

### References

E&E Tech. (N/A, N/A N/A). *AI Phishing Detection*. ai-phishing-detection.

[https://poweronpro.com/services/ai-phishing-detection/?utm\\_source=chatgpt.com](https://poweronpro.com/services/ai-phishing-detection/?utm_source=chatgpt.com)


Graphus AI. (N/A, N/A N/A). *AI Phishing Detection: How to Protect Against Sophisticated Attacks*.

ai-phishing-detection.

[https://www.graphus.ai/ai-phishing-detection/?utm\\_source=chatgpt.com](https://www.graphus.ai/ai-phishing-detection/?utm_source=chatgpt.com)

Ogundairo, O. (2024, August 7). (PDF) *AI-Driven Phishing Detection Systems*. ResearchGate.

Retrieved May 17, 2025, from



[https://www.researchgate.net/publication/382917933\\_AI-Driven\\_Phishing\\_Detection\\_Systems?utm\\_source=chatgpt.com](https://www.researchgate.net/publication/382917933_AI-Driven_Phishing_Detection_Systems?utm_source=chatgpt.com)

Perception Point. (N/A, N/A N/A). *Detecting and Preventing AI-Based Phishing Attacks: 2024*

*Guide*. Perception Point. Retrieved May 17, 2025, from

[https://perception-point.io/guides/ai-security/detecting-and-preventing-ai-based-phishing-attacks-2024-guide/?utm\\_source=chatgpt.com](https://perception-point.io/guides/ai-security/detecting-and-preventing-ai-based-phishing-attacks-2024-guide/?utm_source=chatgpt.com)

Upadhyay, A. (2023, October 23). *Detecting Phishing Attacks with AI | by Akriti Upadhyay*.

Medium. Retrieved May 17, 2025, from

<https://medium.com/@akriti.upadhyay/phishing-detection-met-generative-ai-365b3e89920d>