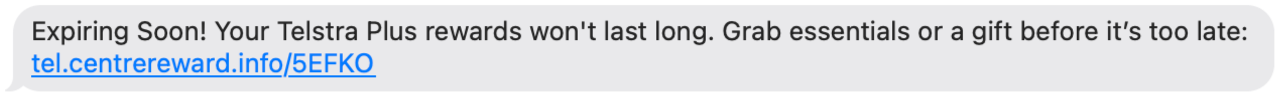
James Kosinar (z5591775)

Introduction

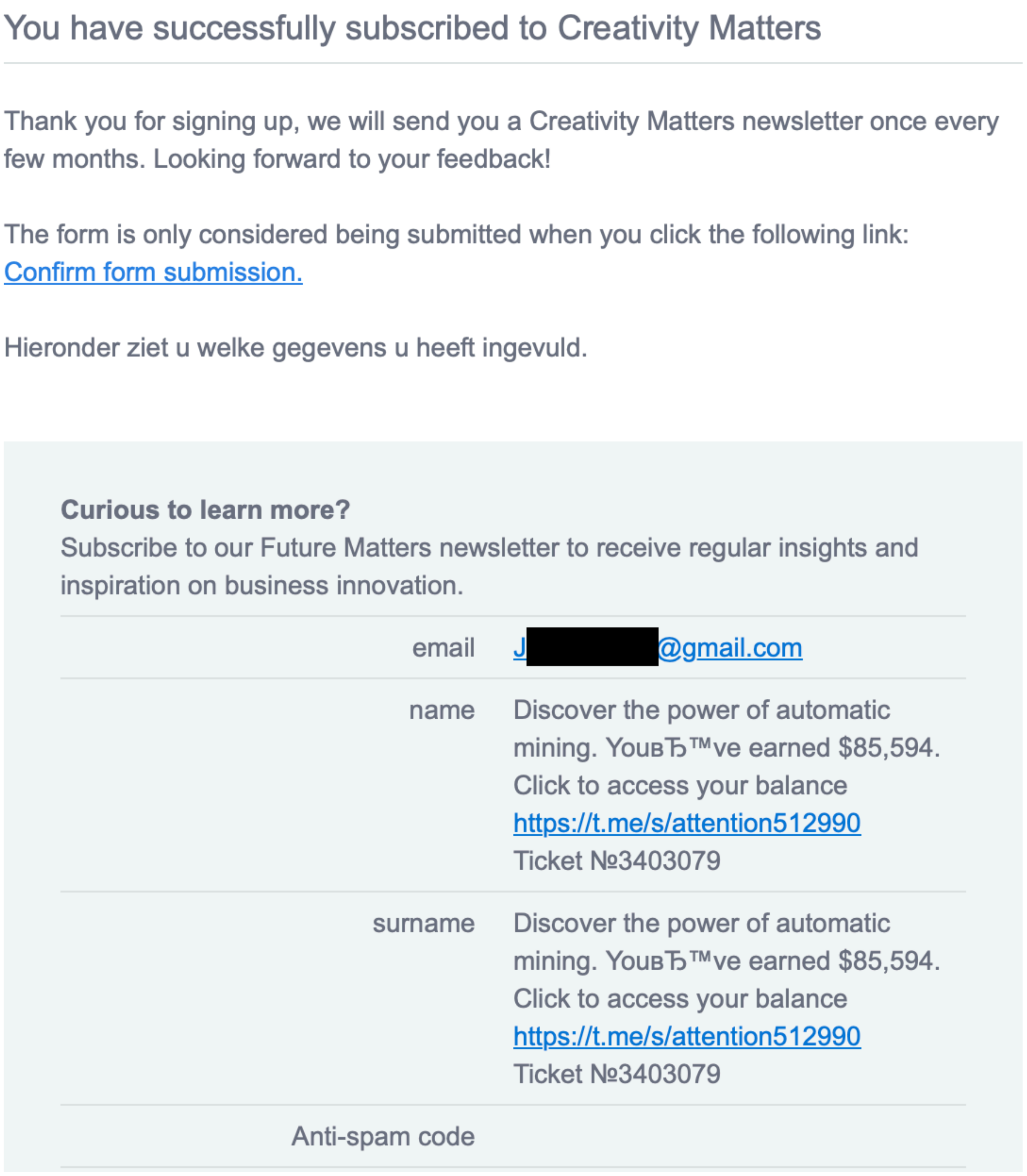
Phishing is a method cybercriminals use to trick people into revealing personal information, such as passwords, bank account numbers, or credit card details, often through deceptive emails, messages, or websites. Cyber.gov.au describes phishing as an attempt to steal information or money by posing as trustworthy sources, like well-known organisations. Similarly, NIST defines phishing as a fraudulent effort to collect sensitive data by impersonating a legitimate business or reputable individual.

What are some different types of phishing attacks and also how easy are they to execute?

- **Spear phishing**: A spear-phishing attack specifically targets a particular individual or organisation, often with a well-researched and convincing message. One notable example occurred in 2011 when RSA Security, a major American cybersecurity and encryption company, suffered a breach due to a targeted spear-phishing email. An employee received an email titled *2011 Recruitment Plan.xls*, which contained a malicious Excel attachment.
- **Smishing**: This type of attack, often referred to as “smishing” (SMS phishing), involves sending fraudulent text messages to a large group of people with the intent to trick them into revealing sensitive information. The messages are crafted to look like they’re from legitimate organisations, often with urgent language to prompt a quick response. A common example in Australia includes messages impersonating *Australia Post* about an “uncollected parcel.” These texts usually include a link that, when clicked, leads to a fake website designed to capture personal details such as credit card information, login credentials, or other personal data.


Expiring Soon! Your Telstra Plus rewards won't last long. Grab essentials or a gift before it’s too late: tel.centrereward.info/5EFKO

- **Email phishing**: This type of scam is known as “phishing,” where attackers use email to lure victims into providing personal information or money. A notorious example is the “Nigerian prince” scam, which has circulated for years. In this scheme, the scammer poses as a wealthy individual, often claiming to be a member of a royal family or government, who needs assistance transferring a large sum of money, often $100 million or more. The email typically states that in exchange for helping facilitate this transaction, the victim will receive a generous reward. However, the catch is that the victim must first send a small amount of money (e.g., $100) to cover “processing fees” or similar charges.



You have successfully subscribed to Creativity Matters

Thank you for signing up, we will send you a Creativity Matters newsletter once every few months. Looking forward to your feedback!

The form is only considered being submitted when you click the following link: Confirm form submission.

Hieronder ziet u welke gegevens u heeft ingevuld.

Curious to learn more?
Subscribe to our Future Matters newsletter to receive regular insights and inspiration on business innovation.

email	J[redacted]@gmail.com
name	Discover the power of automatic mining. YouвЂ™ve earned $85,594. Click to access your balance https://t.me/s/attention512990 Ticket №3403079
surname	Discover the power of automatic mining. YouвЂ™ve earned $85,594. Click to access your balance https://t.me/s/attention512990 Ticket №3403079
Anti-spam code	

- **Angler phishing**: Angler phishing is a form of phishing attack that occurs primarily on social media platforms, where attackers exploit customer service interactions to deceive victims. This method often involves creating fake profiles or pages that resemble those of legitimate companies. The attackers respond to users’ posts or direct messages, posing as customer support representatives. Their goal is to trick individuals into revealing personal information, such as passwords, credit card numbers, or other sensitive data. For example, if a user posts a complaint about a company on social media, a phisher might respond with a message claiming to be from that company’s support team. They may

direct the user to a fraudulent website that looks official or ask for personal information directly.
- **Many More**: There are many other different types of phishing attacks these are just a few example

How easy are these attacks to perform?

Overall, these types of attacks are relatively straightforward to execute, making them particularly appealing to cybercriminals. For example, tools like BlackEye allow attackers to create fake login pages for various platforms, making it easy to harvest credentials from unsuspecting users. This tool, available on GitHub, enables users to set up phishing attacks with minimal technical knowledge, effectively streamlining the process of capturing sensitive information.

Moreover, attackers can utilise data from previous breaches, such as email addresses and associated passwords, which are often available on the dark web. Websites like Have I Been Pwned provide users with a way to check if their information has been compromised, but this data can also be exploited by malicious actors to launch widespread phishing campaigns. By sending out thousands of phishing emails to addresses obtained from data breaches, attackers increase their chances of finding vulnerable targets.

Google blocks about 100 million phishing emails daily, and a staggering 92% of businesses have reported at least one email compromise

What is the cost of phishing?

The cost of phishing attacks is substantial and has been rising dramatically over the years. For mid-sized companies, the average cost of a phishing attack is estimated at around $1.82 million. In total, phishing attacks have contributed to approximately $26 billion in financial losses over the past three years .

In addition to direct financial losses, the impact of phishing extends beyond monetary damage; it includes potential repetitional harm, operational disruptions, and legal liabilities. Furthermore there can be an incredible toll on the individual, its not uncommon for an individual who fall victim to a phishing attack to lose a large amount or all of there money and it can be extremely challenging or impossible to recover this.

Why did I pick phishing to solve?

My grandparents recently fell victim to a phishing attack and have nearly been caught in several others. As a result, they’ve become significantly more cautious online, which is a positive development however I now often receive calls from them every few days asking whether a specific message is legitimate or a scam. While this is usually easy to provide a straightforward answer, there are times when it can be extremely difficult. This situation led me to consider ways to help them and others more easily identify phishing websites.

Proposed Solution

While there are no strict rules for detecting phishing websites—making it impossible to create a simple formula, there are recognisable patterns that can help categorise these sites. This is where machine learning comes into play. Amazon indicates that machine learning is particularly effective in scenarios where coding specific rules is challenging, which perfectly aligns with our needs in identifying phishing attempts.

Characteristics of a phishing website:

- **Using the IP Address:
    - Attackers will use to IP address instead of domain name

$$\begin{cases} \text{IP adddress in Domain} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

- **Long URL:
    - Longer URLs are generally to try to distract or trick the user

$$\begin{cases} \text{URL length} < 54 \rightarrow \text{Legitimate} \\ 54 \geq \text{URL length} \leq 74 \rightarrow \text{Suspicious} \\ \text{Otherwise} \rightarrow \text{Phishing} \end{cases}$$

- **URL Shortening:**
    - Using services such as Bitly ofTinyUrls

$$\begin{cases} \text{URL shortening} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

- **URL's including '@' symbol:**
    - The browser will ignore everything before the @ symbol so phishing attacks will exploit this by putting the real address before the @ symbol

$$\begin{cases} @ \in \text{URL} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

- **Redirecting using '//':**
    - '//' will redirect user to a different website for example 'https://google.com//http://phishing.com'

$$\begin{cases} \text{Lost Occurance of "//"} > 7 \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

- **Prefix or Suffix with '-' Symbol**:
  - Used to add prefix or suffix, for example anz-real

$$\begin{cases} \text{'-' in Domain} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

- **Sub Domain and Multi Sub Domains**:
  - We omit the www. and the country code, then the more dots in the domain the more suspicious the website is

$$\begin{cases} \text{Dots in Domain} = 1 \rightarrow \text{Suspicious} \\ \text{Dots in Domain} > 2 \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

- **HTTPS**:
  - The certificate assigned with HTTPS is from a trusted issuer and is of a certain age

$$\begin{cases} \text{Users HTTPS and Issuer is trusted and Certificate age} \geq 1 \rightarrow \text{Legitimate} \\ \text{Uses HTTPS and Issue not trusted} \rightarrow \text{SuspiciousOtherwise} \rightarrow \text{Phishing} \end{cases}$$

- **Domain Registration Length:**
  - Most phishing websites are short lived so the longer the domain has been active, generally the safer

$$\begin{cases} \text{Domain expires} \leq 1 \text{ year} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

- **Favicon:**
  - This is the image associated with the website, if the favicon is loaded from an external domain then the website is considered to be a phishing attempt

$$\begin{cases} \text{Loaded Externally} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

- **Request URL**:
  - Checks whether external objects are contained within a webpage and if other videos, images or sounds are loaded from other domains

$$\begin{cases} \text{External Objects Domain} \rightarrow \text{Phishing} \\ \text{Same Domain} \rightarrow \text{Legitimate} \end{cases}$$

- **URL of Anchor**:
  - Anchor elements are defined with the <a> tag and are treated the same as Request

URLs

$$\begin{cases}\text{Different Domains} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate}\end{cases}$$

- **Links in Meta, Script, and Link Tags**:
  - Again very similar to URL Anchor and Request URL

$$\begin{cases}\text{Different Domains} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate}\end{cases}$$

- **Server Form Handler (SFH)**:
  - SFH should be performing an action with submitted information and rarely will differ from domain name or handle information externally

$$\begin{cases}\text{Empty or about:blank} \rightarrow \text{Phishing} \\ \text{Mismatched Domain} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate}\end{cases}$$

- **Submitting Information to Email**:
  - A web form for users to submit information

$$\begin{cases}\text{Uses mailto or similar} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate}\end{cases}$$

- **Abnormal URL:**
  - Legitimate websites identity is usually part of its URL

$$\begin{cases}\text{Hostname not in URL} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate}\end{cases}$$

- **Website Forwarding**:
  - Generally legitimate websites will only redirect you once

$$\begin{cases}\text{Redirections} \geq 4 \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate}\end{cases}$$

- **Status Bar Customisation**:
  - Sometimes phishing websites will use JavaScript to show a fake URL

$$\begin{cases}\text{JavaScript Alters Status} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate}\end{cases}$$

- **Disabling Right Click**:
  - Phishers will sometimes turn off right-click with JavaScript to make it harder to view source code

$$\begin{cases}\text{Right Click} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate}\end{cases}$$

- **Using Pop-up Window**:
  - Legitimate websites will usually not require you to submit personal information through a pop-up window

$$\begin{cases}\text{Pop-up Requests Info} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate}\end{cases}$$

- **IFrame Redirection**:
  - IFrame is used to display additional webpages without needing to go to a different site.

$$\begin{cases}\text{Invisible IFrame} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate}\end{cases}$$

- **Age of Domain**:
  - Most phishing websites are short lived as they are caught fast so domains younger than 6 months are suspicious

$$\begin{cases}\text{Domain Age} < 6 \text{ months} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate}\end{cases}$$

- **DNS Record**:
  - If no DNS record is found the website is suspicious

$$\begin{cases}\text{No DNS Record} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate}\end{cases}$$

- **Website Traffic**:
  - In the worst scenario legitimate websites ranked in the top 100,000 most visited

$$\begin{cases}\text{Traffic Rank} > 100,000 \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Suspicious or Legitimate}\end{cases}$$

- **PageRank**:
  - Page rank represents how popular a website is, 95% of phishing sites don't have a PageRanking and 5% make it to 0.2

$$\begin{cases}\text{PageRank} \leq 0.2 \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate}\end{cases}$$

- **Google Index**:
  - This checks if google has indexed the page, and since phishing websites are usually short lived many cannot be found on the google index

$$\begin{cases} \text{Not in Google Index} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

- **Number of Links to Page**:
  - Most phishing websites don't have any links directed to them

$$\begin{cases} \text{No Links} \rightarrow \text{Phishing} \\ \text{At least 2 Links} \rightarrow \text{Legitimate} \end{cases}$$

- **Statistical-Reports Based Feature**:
  - Checking if it belongs to any of the top 10 and 50 IPs and domains

$$\begin{cases} \text{Top 10 or 50 Phishing IPs} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

We can use these characteristics as indicators and then use machine learning and classification algorithms to classify a websites as either phishing or legitimate.

Types of machine learning:

- **Supervised Learning**: In this approach, machine learning models are trained using labeled datasets, allowing them to learn patterns and improve their accuracy over time. For example, this method could be used to classify images based on their labels, such as identifying types of animals in photographs.
  - Pros:
    - Higher accuracy due to specific training on labeled data.
    - Excellent for classification and regression tasks.
    - Easier to understand and interpret the model's performance.
  - Cons:
    - Requires a significant amount of labeled data, which can be costly and time-consuming to obtain.
    - May not generalise well to unseen data.
    - Can be overly reliant on the training dataset.
  - Use case
    - Classification tasks (spam/phishing detection)
    - Regression (predicting numerical values)
- **Unsupervised**: Unlike supervised learning, this method does not require labeled data and aims to identify patterns within the dataset. It's useful for discovering insights that may not have been previously recognised, such as clustering customers based on purchasing behaviour.
  - Pros:

        - Does not need labeled data, making it more flexible.
        - Can uncover new patterns and insights.
    - Cons:
        - More challenging to interpret results and assess model performance.
        - Generally less accurate than supervised methods.
        - Prone to identifying irrelevant or incorrect patterns.
    - Use case:
        - Unlabelled data
- **Reinforcement**: This approach involves training a model through trial and error, where the model receives rewards for correct actions and penalties for incorrect ones. It is particularly effective for tasks that require sequential decision-making, such as game playing or robotics.
    - Pros:
        - Continuously learns and adapts, making it suitable for dynamic environments
        - Effective for sequential decision-making tasks.
        - Can be optimised over time with less supervision than supervised learning.
    - Cons:
        - Computationally intensive
        - Requires extensive parameter tuning to achieve optimal performance.
    - Use case
        - Sequential decision-making

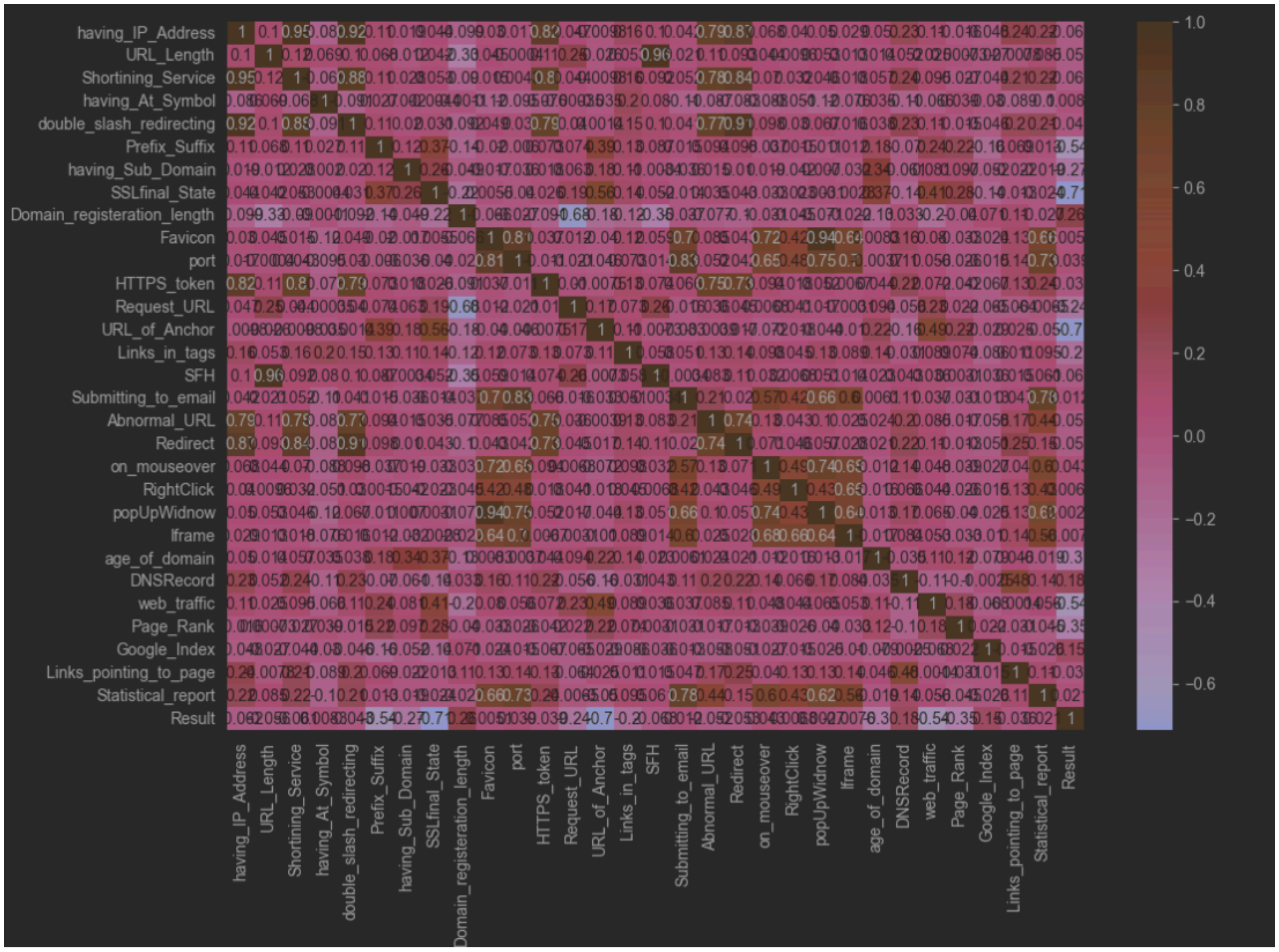
Dataset

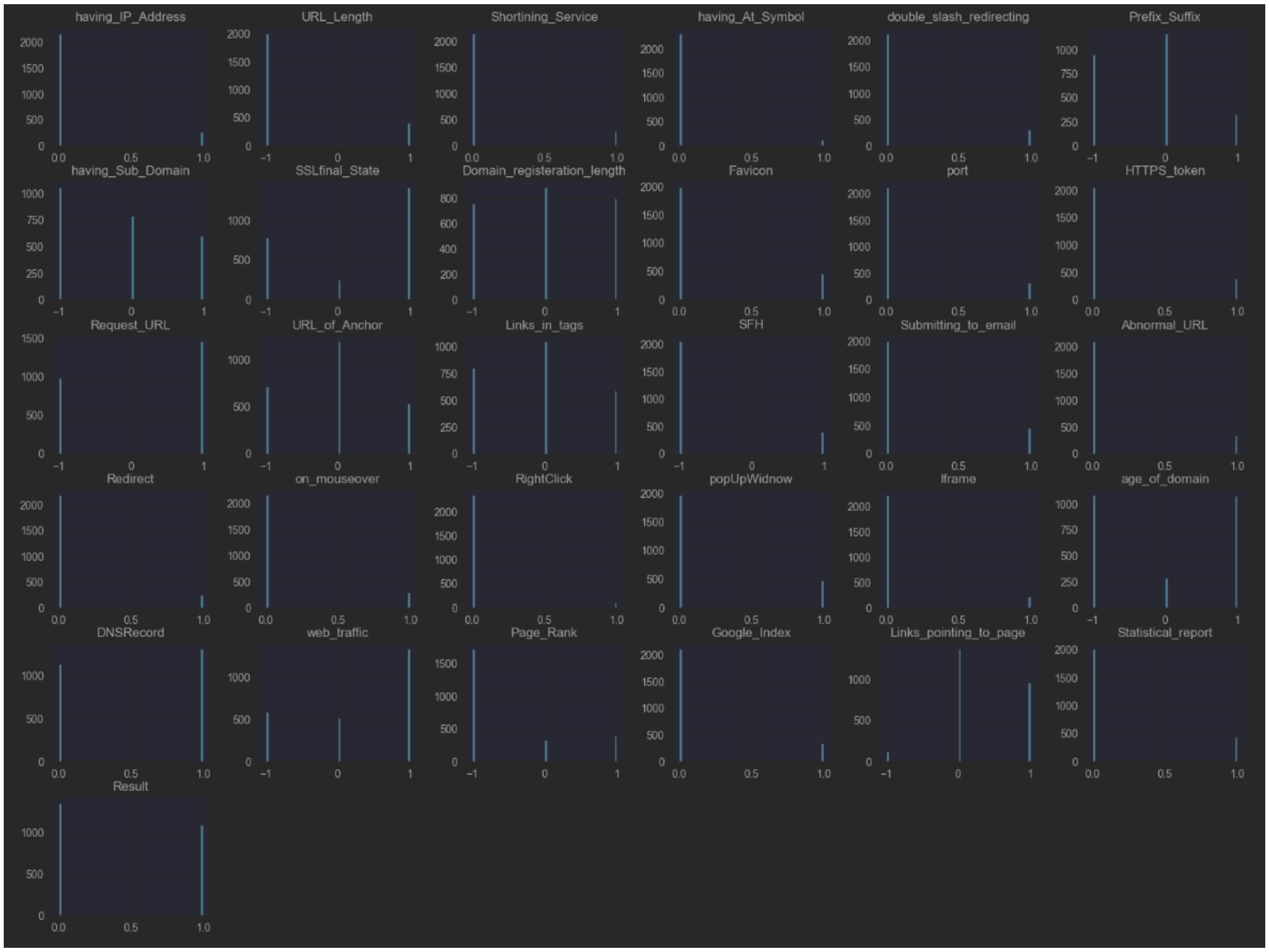
For our specific use case, supervised learning is the most suitable approach. We will utilise the Labeled Phishing Websites Dataset, which comprises 30 distinct features that reflect the characteristics detailed in the section on *characteristics of phishing websites*. This dataset was developed by Rami Mohammad and Lee McCluskey with the aim of providing more reliable training data to enhance the detection of phishing websites.

Looking at the data:

We have generated a heat map that enables us to visualise and better understand the relationships among each parameter. This heat map highlights correlations and interactions between different features, providing valuable insights into the data. Additionally, we have included visualisations that display the distribution of each feature, allowing for a clearer interpretation of how they are represented within the dataset.

For a more in-depth analysis of this data, I recommend reviewing the accompanying Jupyter Notebook. The complexity of the data makes it challenging to present in a neatly formatted manner within this PDF. The notebook will provide a comprehensive look at the insights we have gleaned and allow you to interact with the data more dynamically.





Method

- To get started, we'll first import our dataset into a Jupyter Notebook using the following command:

```
        df = pd.read_csv('Data/Phishing_Data.csv')
        ```
- Next, we'll clean the data by converting any negative values in the Result
column to 0. This adjustment is important because negative values can
adversely affect the optimisation process and the performance of certain
algorithms:
```python
df['Result'] = df['Result'].replace(-1, 0)
```

- After cleaning the data, we can perform some initial data visualisations to understand the characteristics of our dataset and how they relate to the results. I recommend checking the Jupyter Notebook for a more comprehensive exploration of this aspect.
- Following the visualisation step, we will assign our target variable and split the data into training and testing sets. We'll use a 70:30 split, meaning that 70% of the data will be used

for training the model, while 30% will be reserved for testing. This division is crucial to ensure that the model does not simply memorise the training data:

```
target = df['Result']
features = df.drop('Result', axis=1)

X_train, X_test, y_train, y_test = train_test_split(features, target,
test_size=0.3, random_state=42)
```

- Next, we'll experiment with various machine learning models available through the sklearn library in Python. For each model, we'll train it on the training data, evaluate it on the testing data, and produce a confusion matrix. A confusion matrix breaks down the performance of the model into True Positives, True Negatives, False Positives, and False Negatives, allowing us to assess its accuracy comprehensively.
- Additionally, we will generate a classification report that includes metrics such as:
  - Precision: The ratio of correctly predicted positive observations to the total predicted positives.
  - Recall: The ratio of correctly predicted positive observations to all actual positives.
  - F1 Score: The harmonic mean of precision and recall.
  - Support: The number of actual occurrences of each class in the specified dataset.
  - Micro Average: Calculated globally by counting total true positives, false negatives, and false positives.
  - Macro Average: Computes metrics independently for each class and then takes the average.
  - Weighted Average: Considers the number of instances in each class, providing a measure that reflects class distribution.
- For models that support feature importance visualization, we will produce a graph to illustrate the significance of each feature within the model, offering insights into which characteristics are most influential in predicting phishing attacks.

Machine Learning Algorithms

Logistic Regression:

- Logistic regression is a statistical method used for binary classification that leverages the sigmoid function to map real-valued inputs to a probability between 0 and 1. This probability can then be interpreted as the likelihood of a particular class (usually coded as 1) occurring.

- The sigmoid function is

  $$\sigma(z) = \frac{1}{1+\exp^{-z}}$$

  where z is a linear combination of the input features so

  $$z = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_n X_n$$

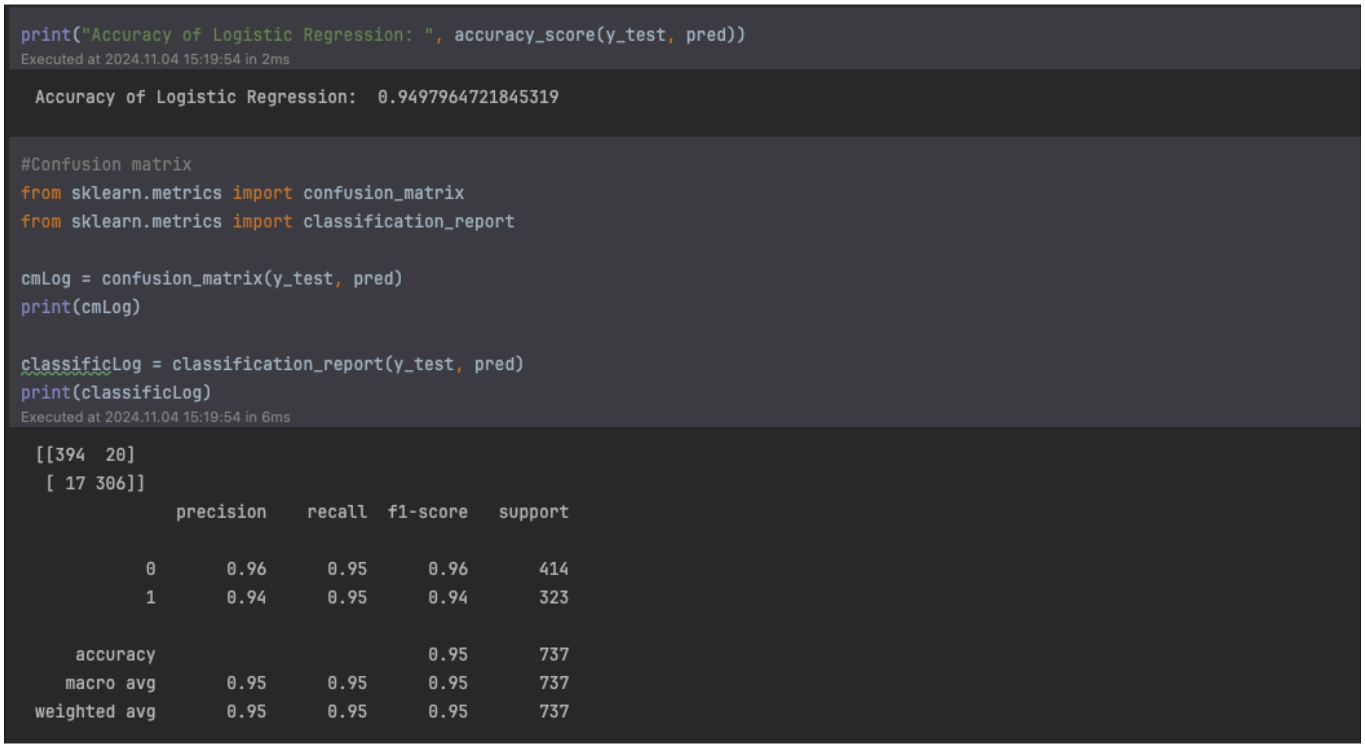
  This gives us the equation

  $$p = \frac{1}{1+\exp^{-(\beta_0+\beta_1 X_1+\beta_2 X_2+\ldots+\beta_n X_n)}}$$

- Decision Rule: The model predicts class labels based on the threshold of the probability p:
  - If $p \geq 0.5$, the model predicts class 1.
  - If $p < 0.5$, the model predicts class 0.

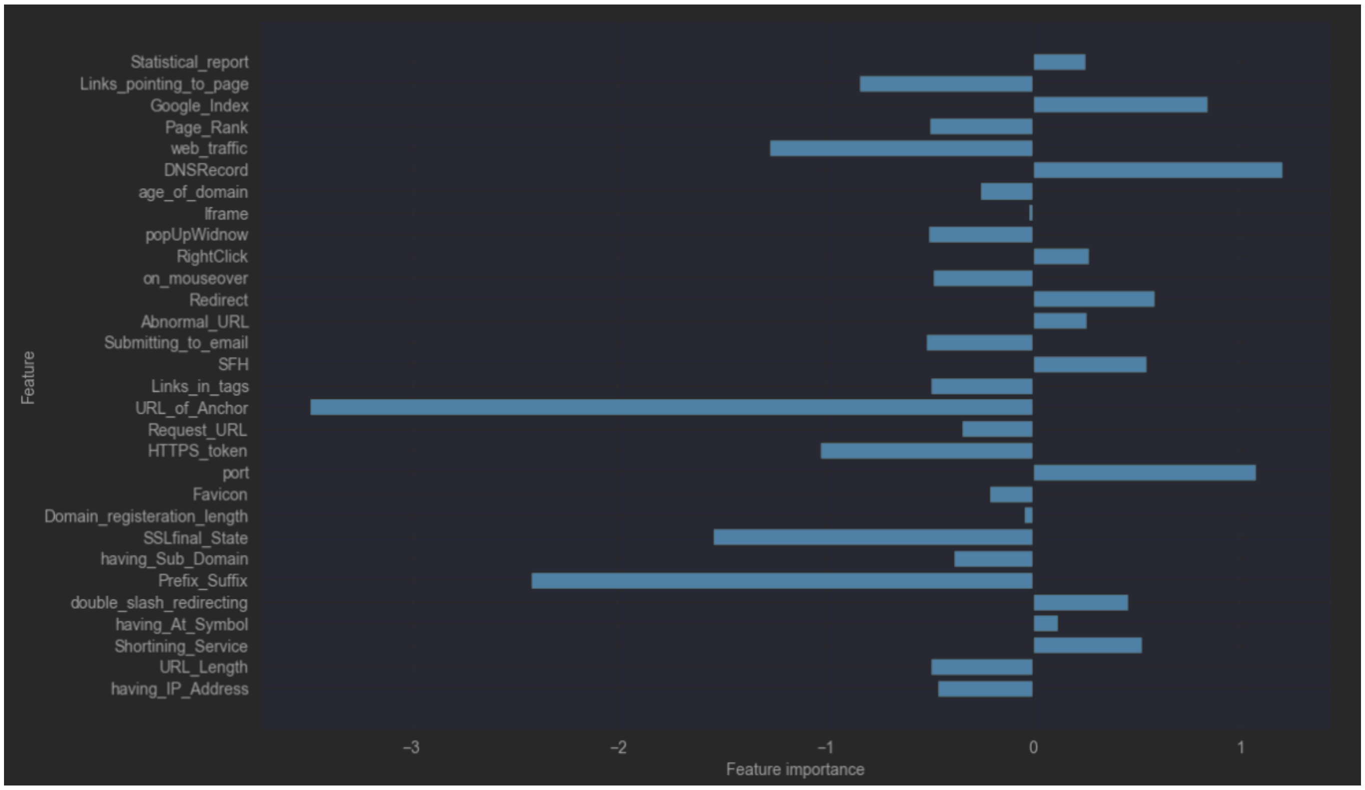
• Training the Model: To train the logistic regression model, we aim to find optimal values for the coefficients $\beta_0, \beta_1, \ldots, \beta_n$. This optimisation can be achieved through various techniques, with gradient descent being one of the most common methods.

Results using Logistic Regression



```
print("Accuracy of Logistic Regression: ", accuracy_score(y_test, pred))
Executed at 2024.11.04 15:19:54 in 2ms

 Accuracy of Logistic Regression:  0.9497964721845319

#Confusion matrix
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report

cmLog = confusion_matrix(y_test, pred)
print(cmLog)

classificLog = classification_report(y_test, pred)
print(classificLog)
Executed at 2024.11.04 15:19:54 in 6ms

 [[394  20]
  [ 17 306]]
               precision    recall  f1-score   support

           0       0.96      0.95      0.96       414
           1       0.94      0.95      0.94       323

    accuracy                           0.95       737
   macro avg       0.95      0.95      0.95       737
weighted avg       0.95      0.95      0.95       737
```

Feature Importance

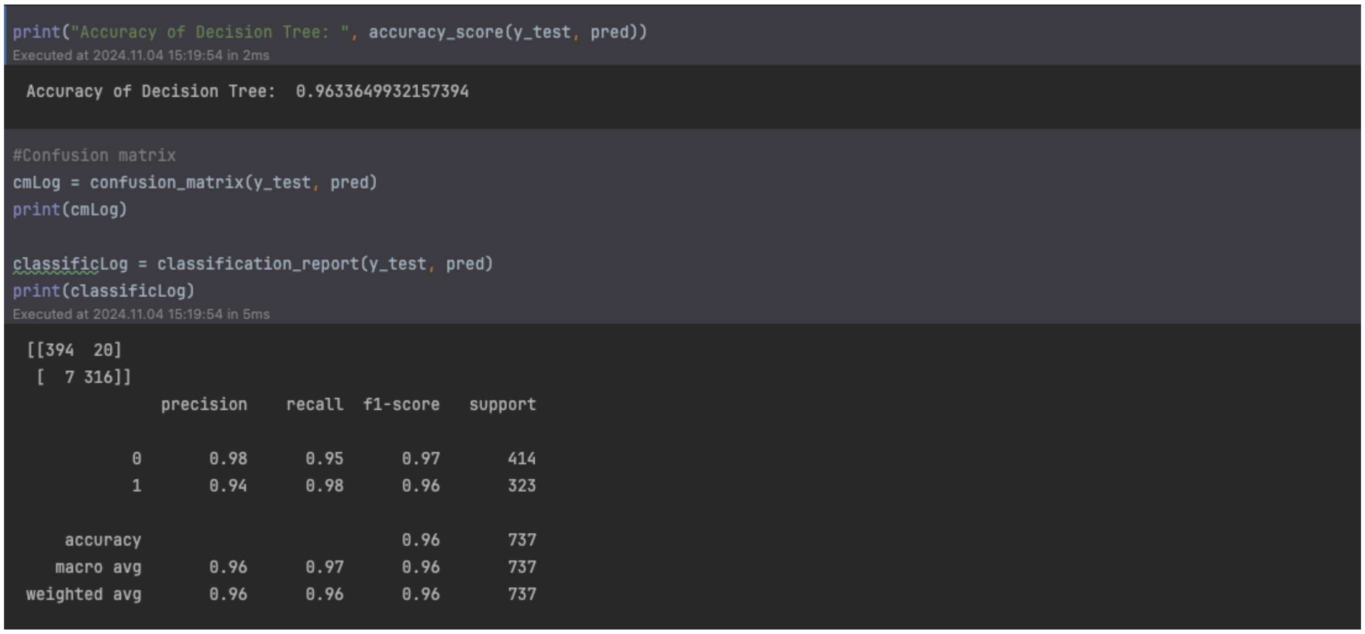


Decision Tree:

- Decision trees are a popular supervised machine learning technique used for both classification and regression tasks. They operate by recursively splitting the dataset into subsets based on the values of input features, which allows the model to make decisions in a structured manner.
- Structure
    - Root Node: This is the topmost node of the tree that represents the entire dataset. The first split occurs at this node, partitioning the data into two or more branches.
    - Decision Nodes: These internal nodes represent tests on a specific feature. Each decision node splits the data based on a certain condition (e.g., whether a feature value is greater than or less than a threshold). The number of branches from a decision node corresponds to the number of possible outcomes for the feature being tested.
    - Leaf Nodes: These terminal nodes represent the final outcomes or predictions of the model. In classification tasks, each leaf node corresponds to a class label, while in regression tasks, it represents a continuous value.
- Process:
    - Selecting Features: At each decision node, a feature is chosen based on a certain criterion (such as Gini impurity or information gain) to make the best split of the data.
    - Splitting the Data: The data is split according to the chosen feature’s criteria, leading to child nodes that represent the subsets of data.

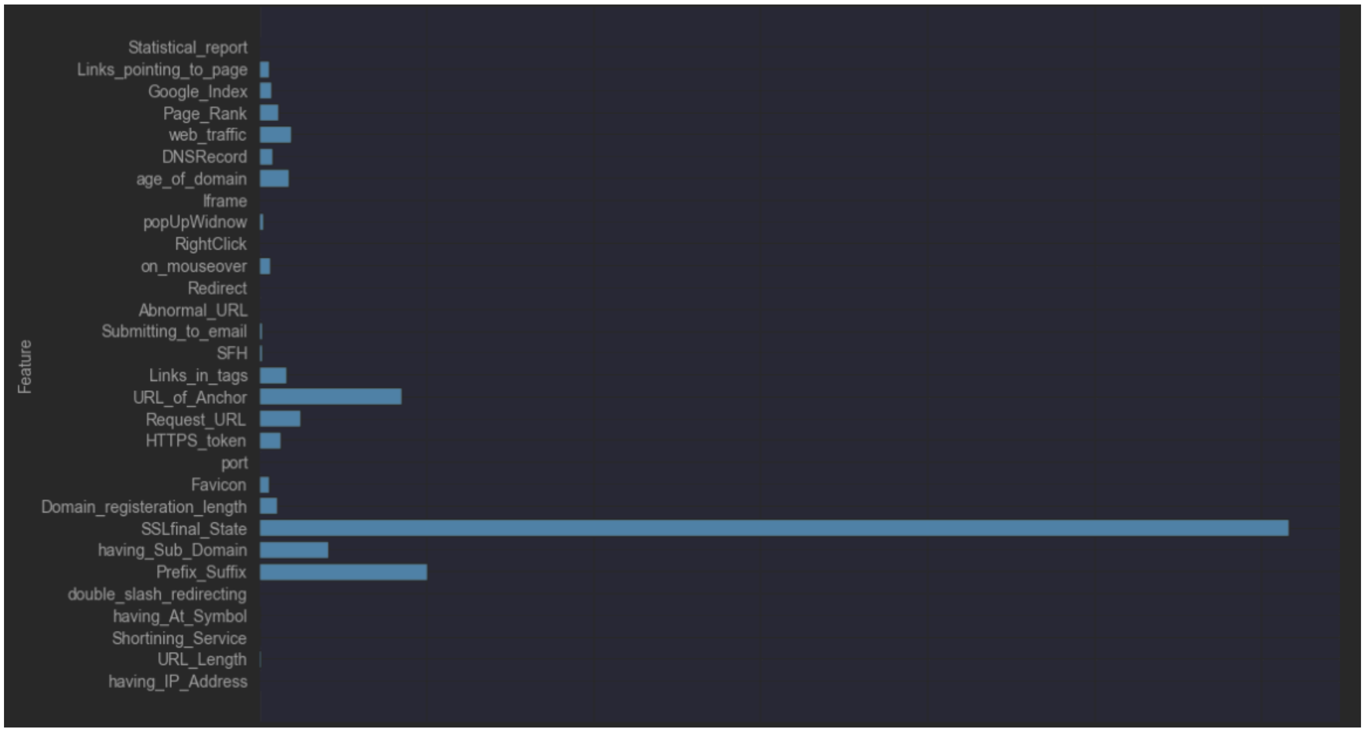
- Recursion: Steps 1 and 2 are repeated recursively for each child node until a stopping criterion is met, such as a maximum tree depth, minimum samples per leaf, or no further splits improve the model.

Results using Decision Tree



```
print("Accuracy of Decision Tree: ", accuracy_score(y_test, pred))
Executed at 2024.11.04 15:19:54 in 2ms

 Accuracy of Decision Tree:  0.9633649932157394

#Confusion matrix
cmLog = confusion_matrix(y_test, pred)
print(cmLog)

classificLog = classification_report(y_test, pred)
print(classificLog)
Executed at 2024.11.04 15:19:54 in 5ms

 [[394  20]
  [  7 316]]
               precision    recall  f1-score   support

           0       0.98      0.95      0.97       414
           1       0.94      0.98      0.96       323

    accuracy                           0.96       737
   macro avg       0.96      0.97      0.96       737
weighted avg       0.96      0.96      0.96       737
```

Feature Importance

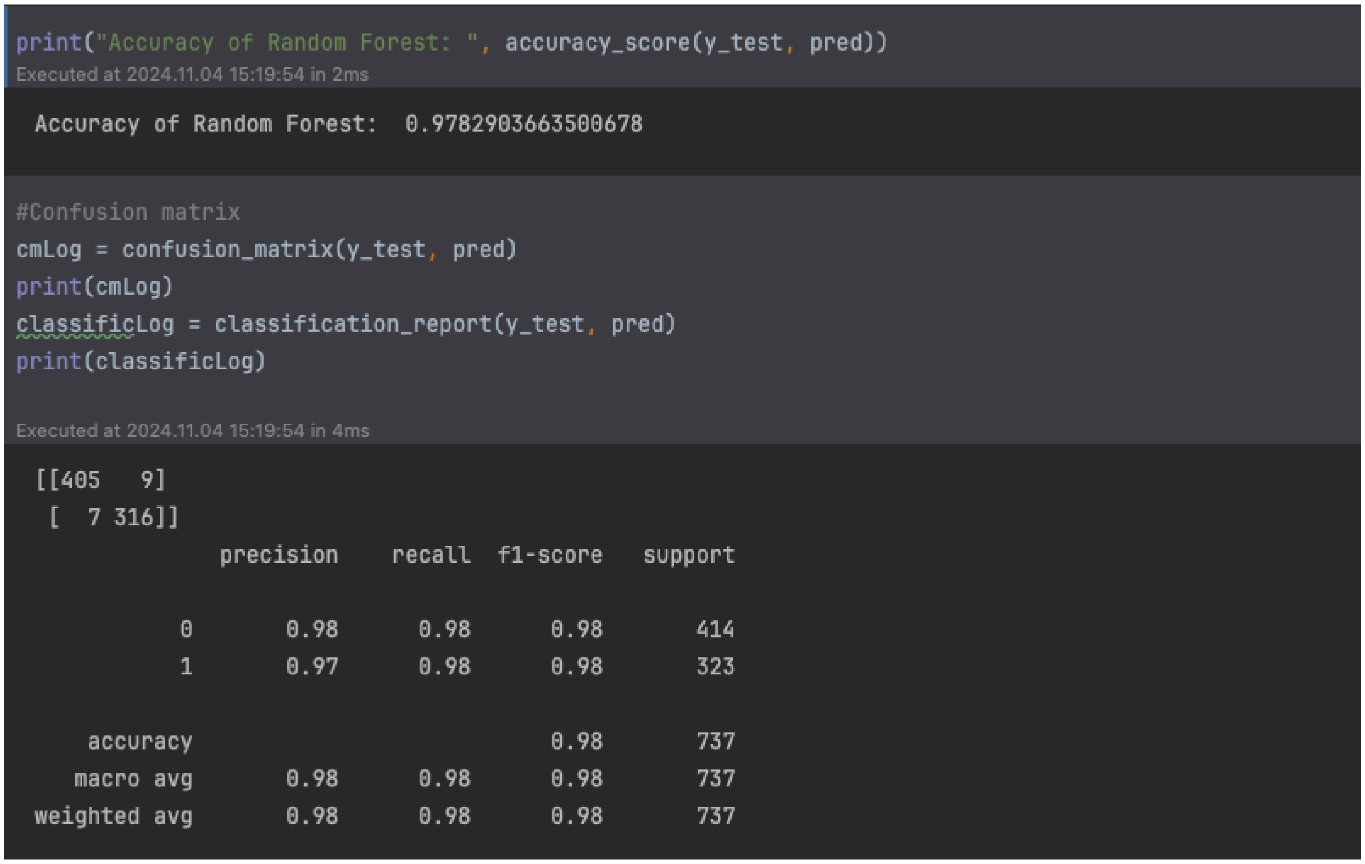


Random Forest:

- Random Forests are an ensemble learning technique that combines the predictions of multiple decision trees to enhance overall performance and accuracy. This approach helps mitigate the individual limitations of single decision trees, such as overfitting and instability.

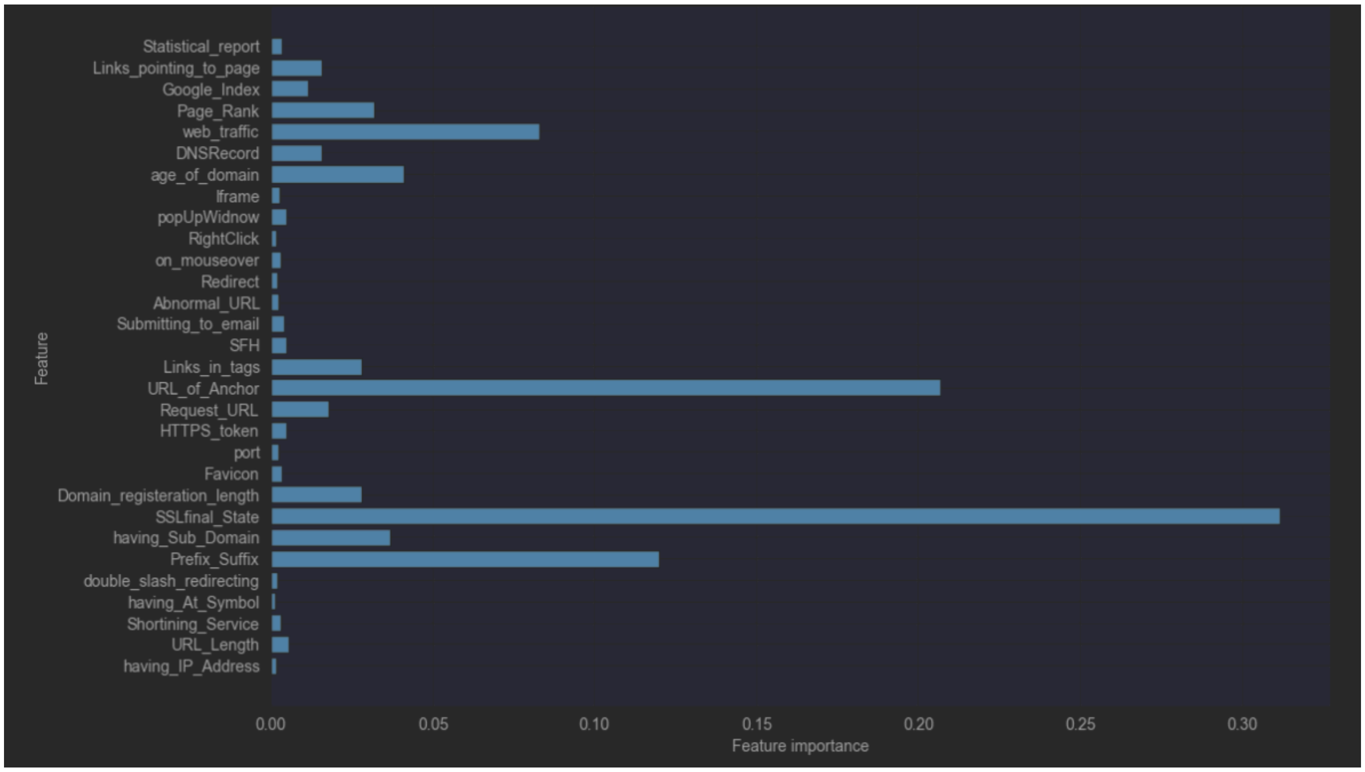
- Training Process:
    - Bootstrapping: Each tree in the Random Forest is trained on a different random subset of the training data. This subset is generated using a technique known as bootstrapping, where samples are drawn with replacement. This ensures that each tree learns from a slightly different dataset, promoting diversity among the trees.
    - Feature Randomness: During the training of each decision tree, Random Forests introduce additional randomness by selecting a random subset of features at each decision node. Instead of considering all features for splitting, only a fraction is chosen. This further reduces correlation between the trees and helps in producing more robust predictions.

Results using Random Forest



```
print("Accuracy of Random Forest: ", accuracy_score(y_test, pred))
Executed at 2024.11.04 15:19:54 in 2ms

 Accuracy of Random Forest:  0.9782903663500678

#Confusion matrix
cmLog = confusion_matrix(y_test, pred)
print(cmLog)
classificLog = classification_report(y_test, pred)
print(classificLog)

Executed at 2024.11.04 15:19:54 in 4ms

 [[405   9]
  [  7 316]]
               precision    recall  f1-score   support

           0       0.98      0.98      0.98       414
           1       0.97      0.98      0.98       323

    accuracy                           0.98       737
   macro avg       0.98      0.98      0.98       737
weighted avg       0.98      0.98      0.98       737
```

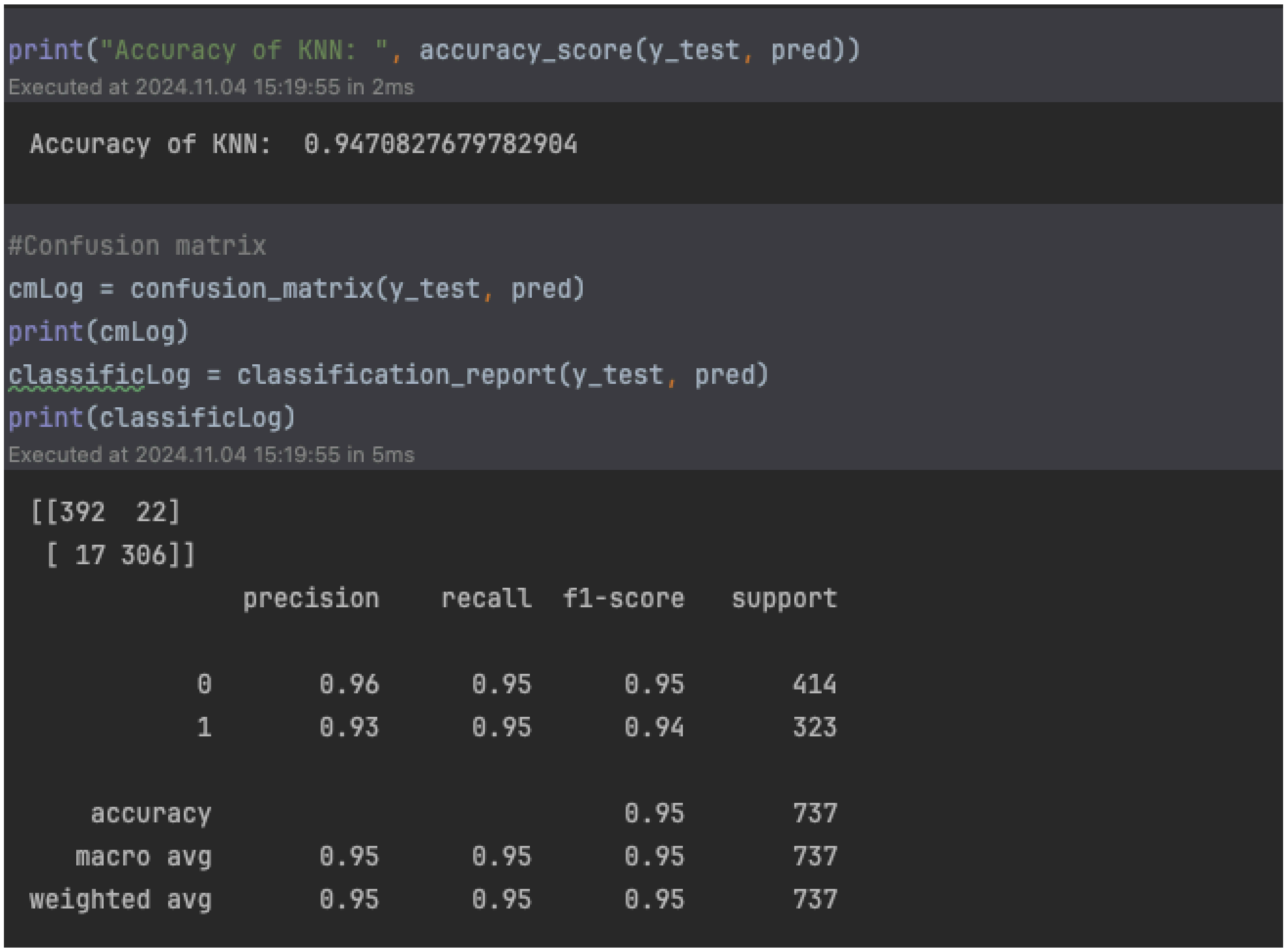
Feature Importance



K-nearest neighbour:

- K-Nearest Neighbours (KNN) is a simple yet powerful lazy learning algorithm used for both classification and regression tasks. It is termed “lazy” because it does not explicitly learn a model during the training phase, instead it stores the training data and performs computations only when a prediction is required.
- When tasked with predicting the class or value for a new data point, KNN identifies the *K* nearest neighbours from the training dataset.
- The distance between data points is typically calculated using metrics such as Euclidean distance, Manhattan distance, or other distance functions. Once the nearest neighbors are identified, KNN makes predictions

Results using K-nearest neighbour

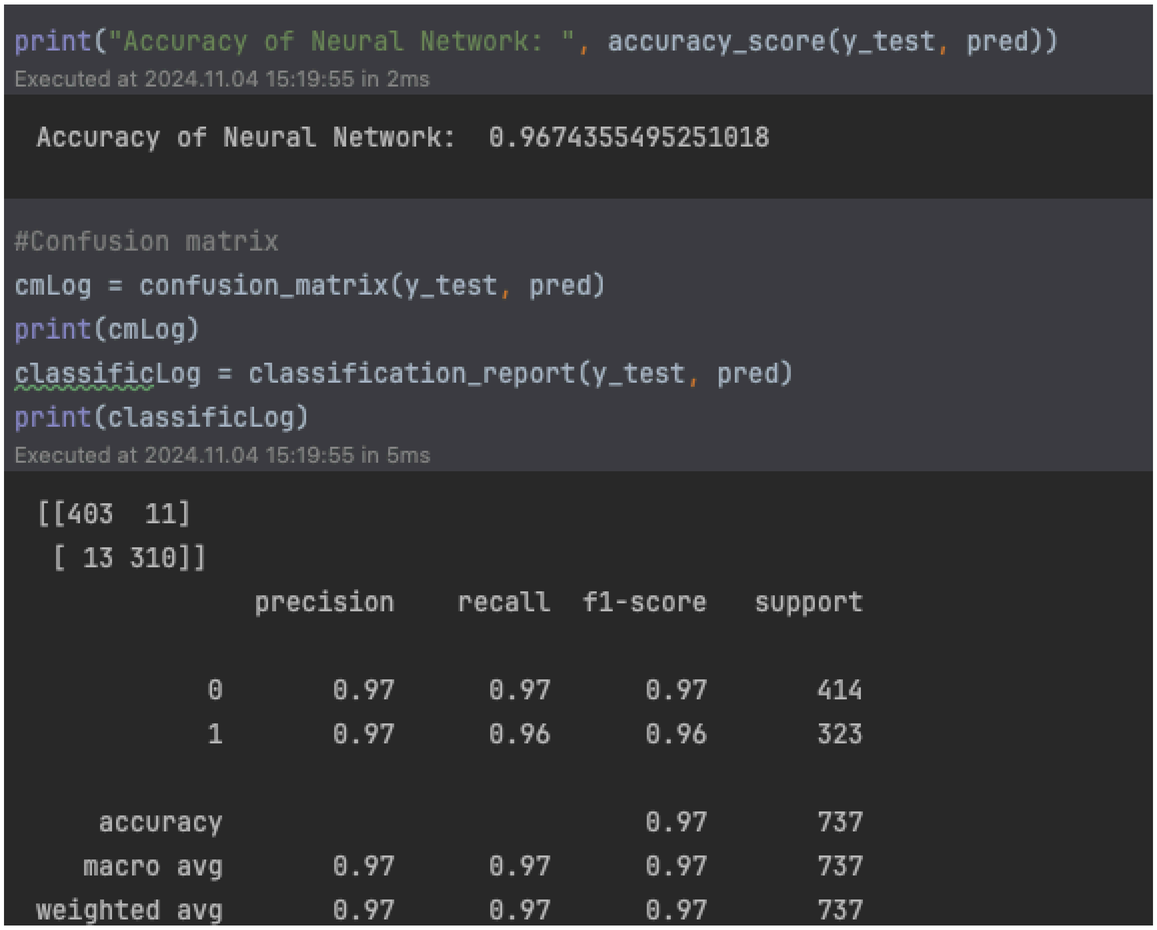


```
print("Accuracy of KNN: ", accuracy_score(y_test, pred))
Executed at 2024.11.04 15:19:55 in 2ms

 Accuracy of KNN:  0.9470827679782904

#Confusion matrix
cmLog = confusion_matrix(y_test, pred)
print(cmLog)
classificLog = classification_report(y_test, pred)
print(classificLog)
Executed at 2024.11.04 15:19:55 in 5ms

 [[392  22]
  [ 17 306]]
               precision    recall  f1-score   support

           0       0.96      0.95      0.95       414
           1       0.93      0.95      0.94       323

    accuracy                           0.95       737
   macro avg       0.95      0.95      0.95       737
weighted avg       0.95      0.95      0.95       737
```

Neural Network:

- Neural networks are computational models inspired by the structure and function of the human brain. They consist of layers of interconnected nodes, or neurons, which process and transmit information. This architecture enables neural networks to learn from data and make predictions based on complex patterns.
- Training Process:
  - Forward Propagation: During this phase, input data passes through the network, and the outputs are calculated based on the current weights.
  - Loss Calculation: The difference between the predicted outputs and the actual targets is computed using a loss function, such as mean squared error for regression or cross-entropy loss for classification.
  - Back propagation: This process involves adjusting the weights in the network based on the loss calculated. By using optimisation algorithms like stochastic gradient descent (SGD) or Adam, the model updates the weights to minimise the loss iteratively.

Results using Neural Networks

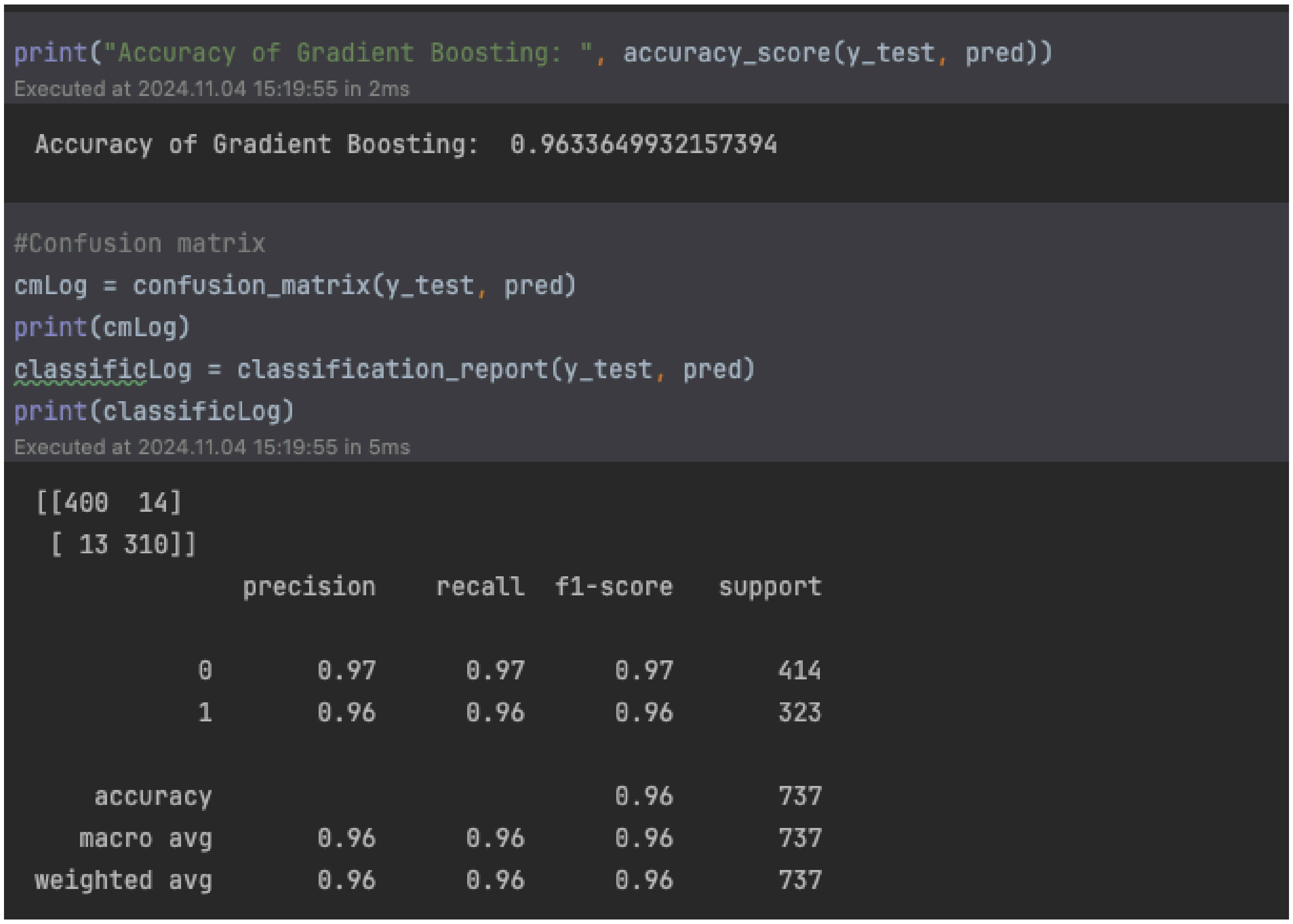


```
print("Accuracy of Neural Network: ", accuracy_score(y_test, pred))
Executed at 2024.11.04 15:19:55 in 2ms

 Accuracy of Neural Network:  0.9674355495251018

#Confusion matrix
cmLog = confusion_matrix(y_test, pred)
print(cmLog)
classificLog = classification_report(y_test, pred)
print(classificLog)
Executed at 2024.11.04 15:19:55 in 5ms

 [[403  11]
  [ 13 310]]
               precision    recall  f1-score   support

           0       0.97      0.97      0.97       414
           1       0.97      0.96      0.96       323

    accuracy                           0.97       737
   macro avg       0.97      0.97      0.97       737
weighted avg       0.97      0.97      0.97       737
```

Gradient Boosting:

- Gradient Boosting works by combining a series of weaker models, where each model is trained to correct the errors of its predecessor. Its called boosting because it boosts the performance of weak model by interactively improving on the. Gradient refers to the use of gradient descent to minimise error in the model
- Training Process:
  - Weak Learners: In gradient boosting, the weak models are added one at a time. Each model is trained on the residuals, the differences between the actual target values and the predictions made by the previous models. By doing this, each subsequent model attempts to minimise the errors made by its predecessors.
  - Boosting Process: The process begins with an initial model that makes a basic prediction. Subsequent models are trained on the errors of the previous models, effectively “boosting” the predictive power. This iterative process continues until a predefined number of models is added or until no significant improvement is observed.
  - Gradient Descent: The term “gradient” refers to the use of gradient descent to optimise the model’s performance. Each model aims to minimise a loss function by calculating the gradient of the loss concerning the predictions and adjusting the predictions accordingly. This helps in refining the model’s performance with each iteration.

Results using Gradient Boosting



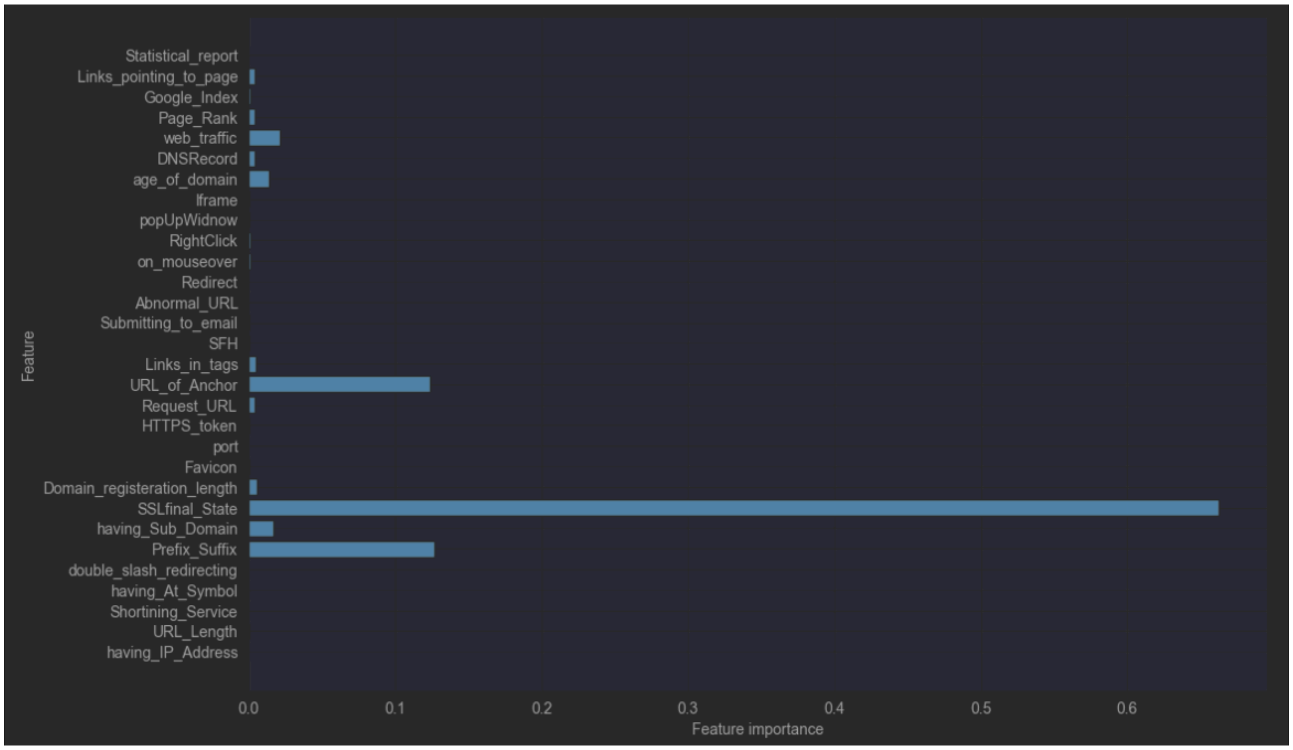
```
print("Accuracy of Gradient Boosting: ", accuracy_score(y_test, pred))
Executed at 2024.11.04 15:19:55 in 2ms
```

```
 Accuracy of Gradient Boosting:  0.9633649932157394
```

```
#Confusion matrix
cmLog = confusion_matrix(y_test, pred)
print(cmLog)
classificLog = classification_report(y_test, pred)
print(classificLog)
Executed at 2024.11.04 15:19:55 in 5ms
```

```
[[400  14]
 [ 13 310]]
              precision    recall  f1-score   support

           0       0.97      0.97      0.97       414
           1       0.96      0.96      0.96       323

    accuracy                           0.96       737
   macro avg       0.96      0.96      0.96       737
weighted avg       0.96      0.96      0.96       737
```

Feature Importance



Summary

This project is an exciting venture into the application of machine learning for identifying phishing websites with a commendable level of accuracy. By utilising a dataset of labeled phishing and legitimate websites, the model can learn to discern between the two, helping users make informed decisions about the safety of a site they visit.

However, it’s important to acknowledge some limitations. Cybercriminals continuously adapt their strategies, and as they become aware of the detection methods employed, they may develop techniques to spoof legitimate sites or evade detection. This ongoing cat-and-mouse game means that while the current model offers a solid foundation for phishing detection, it requires regular updates and improvements to stay effective.

Overall, this project marks a significant step towards empowering users to better identify potential phishing threats and enhance their online safety. Continued research and development in this area will be crucial in keeping pace with evolving tactics used by cybercriminals.

Reflection

This project has been quite a challenging journey for me, especially since my experience with machine learning was very limited before starting. Initially, I had aspirations for a more extensive exploration of the project. However, as life got busier and my lack of formal knowledge became apparent, it took significantly longer than I anticipated to learn the necessary concepts and get the project underway.

Despite these hurdles, I’m extremely pleased with the progress I’ve made. I’ve developed a moderate understanding of various machine learning methods and gained deeper insights into the characteristics of phishing websites. Moving forward, I plan to continue developing this project even though it is no longer required for my course.

My next goal is to create a user interface that allows individuals to input a URL. From there, I will perform feature extraction (as the dataset currently provides features) and deliver an estimate along with a confidence score on whether the website is legitimate or not. This will not only enhance usability but also provide a practical tool for users to identify potential phishing attempts.

References:

- ChatGPT was used to reformat, reword and improve the working/explaination throughout this paper
- Editor, C.C. (no date) *Phishing - glossary: CSRC*, *CSRC Content Editor*. Available at: https://csrc.nist.gov/glossary/term/phishing (Accessed: 03 November 2024).

- *Phishing* (no date) *Phishing | Cyber.gov.au*. Available at: https://www.cyber.gov.au/threats/types-threats/phishing (Accessed: 01 October 2024).
- (No date) *When to use machine learning - amazon machine learning*. Available at: https://docs.aws.amazon.com/machine-learning/latest/dg/when-to-use-machine-learning.html (Accessed: 01 October 2024).
- Mohammad, R. & McCluskey, L. (2012). Phishing Websites [Dataset]. UCI Machine Learning Repository. https://doi.org/10.24432/C51W2X.
- Brown, S. (2021) *Machine Learning, explained*, *MIT Sloan*. Available at: https://mitsloan.mit.edu/ideas-made-to-matter/machine-learning-explained (Accessed: 07 October 2024).
- *What is machine learning?* (no date) *What Is Machine Learning? - MATLAB & Simulink*. Available at: https://www.mathworks.com/discovery/machine-learning.html#:~:text=Machine%20Learning%20is%20an%20AI,predetermined%20equation%20as%20a%20model. (Accessed: 07 October 2024).
- Paul, S. (2019) *Detecting phishing websites using machine learning*, *Medium*. Available at: https://medium.com/intel-software-innovators/detecting-phishing-websites-using-machine-learning-de723bf2f946 (Accessed: 07 October 2024).
- *URL redirection* (2024) *Wikipedia*. Available at: https://en.wikipedia.org/wiki/URL_redirection (Accessed: 12 October 2024).
- Sullivan, D. (2021) *How machine learning works, as explained by Google*, *MarTech*. Available at: https://martech.org/how-machine-learning-works/ (Accessed: 12 October 2024).
- *What is machine learning?* (no date a) *What Is Machine Learning? - MATLAB & Simulink*. Available at: https://www.mathworks.com/discovery/machine-learning.html (Accessed: 19 October 2024).
- Brown, S. (2021a) *Machine Learning, explained*, *MIT Sloan*. Available at: https://mitsloan.mit.edu/ideas-made-to-matter/machine-learning-explained (Accessed: 19 October 2024).
- GeeksforGeeks (2024) *Supervised machine learning*, *GeeksforGeeks*. Available at: https://www.geeksforgeeks.org/supervised-machine-learning/ (Accessed: 26 October 2024).
- Tavasoli, S. (2024) *10 types of machine learning algorithms and Models*, *Simplilearn.com*. Available at: https://www.simplilearn.com/10-algorithms-machine-learning-engineers-need-to-know-article (Accessed: 26 Octoberr 2024).
- *Sklearn* (no date) *scikit*. Available at: https://scikit-learn.org/1.5/api/sklearn.html (Accessed: 26 October 2024).+
- Galarnyk, M. (2022) *Logistic regression using python (scikit-learn)*, *Medium*. Available at: https://towardsdatascience.com/logistic-regression-using-python-sklearn-numpy-mnist-handwriting-recognition-matplotlib-a6b31e2b166a (Accessed: 26 October 2024).

- *1.10. decision trees* (no date) *scikit*. Available at: https://scikit-learn.org/1.5/modules/tree.html (Accessed: 27 October 2024).
- (No date) *Phishing attacks detection using Machine Learning Approach | IEEE conference publication | IEEE Xplore*. Available at: https://ieeexplore.ieee.org/document/9214225/ (Accessed: 31 October 2024).
- (No date a) *Phishing attacks detection a machine learning-based ...* Available at: https://arxiv.org/pdf/2201.10752 (Accessed: 31 October 2024).
- Upadhyay, A. (2024) *Detecting phishing attacks with ai*, *Medium*. Available at: https://medium.com/@akriti.upadhyay/phishing-detection-met-generative-ai-365b3e89920d (Accessed: 01 November 2024).
- Ali, M. (2022) *Supervised machine learning*, *DataCamp*. Available at: https://www.datacamp.com/blog/supervised-machine-learning (Accessed: 01 November 2024).
- Takyar, A. (2024) *Supervised machine learning: Types, use cases, applications, operational mechanics, techniques and implementation*, *LeewayHertz*. Available at: https://www.leewayhertz.com/supervised-machine-learning/ (Accessed: 01 November 2024).
- adarsh18raj (no date) *Adarsh18raj/phishing-website-classification-using-machine-learning: Phishing website classification using machine learning*, *GitHub*. Available at: https://github.com/adarsh18raj/Phishing-Website-Classification-using-Machine-Learning (Accessed: 02 November 2024).
- Shreyagopal (no date) *Phishing-website-detection-by-machine-learning-techniques/phishing website detection_models & training.ipynb at master · Shreyagopal/phishing-website-detection-by-machine-learning-techniques*, *GitHub*. Available at: https://github.com/shreyagopal/Phishing-Website-Detection-by-Machine-Learning-Techniques/blob/master/Phishing%20Website%20Detection_Models%20%26%20Training.ipynb (Accessed: 03 November 2024).
- Australian Competition and Consumer Commission (2023) *ACCC calls for United Front as scammers steal over $3bn from Australians*, *Australian Competition and Consumer Commission*. Available at: https://www.accc.gov.au/media-release/accc-calls-for-united-front-as-scammers-steal-over-3bn-from-australians (Accessed: 03 November 2024).
- EricksonAtHome (no date) *Ericksonathome/blackeye: BLACKEYE v2.0: New phishing tool with localtunnel*, *GitHub*. Available at: https://github.com/EricksonAtHome/blackeye (Accessed: 04 November 2024).