LOVELY
PROFESSIONAL
UNIVERSITY

Transforming Education Transforming India

LOVELY PROFESSIONA UNIVERSITY

PHAGWARA(PUNJAB)

PROJECT – WEBSITE PHISHING

Name – Shanu kumar

Reg.no – 12016533

Section – K20RK

Course code – INT354

Link : https://colab.research.google.com/drive/1kRbpiBpkWCSX2OXY49zeyrKDcoRK-_Rj?usp=share_link

ABSTRACT : -

Internet and cloud technology advancements in recent years have significantly increased electronic trading, or consumer-to-consumer online transactions. The resources of an enterprise are harmed by this growth, which permits unauthorized access to sensitive information about users. One well-known attack that deceives users into accessing malicious content and giving up their information is phishing. Most phishing websites have the same website interface and uniform resource locator (URL) as the legitimate websites. There have been many suggested methods for identifying phishing, including blacklists, heuristics, etc. However, the number of victims is rising exponentially as a result of inadequate security technologies. Phishing attacks are more likely to succeed on the Internet because of its anonymous and uncontrollable nature. Existing research demonstrates that the phishing detection system's performance is constrained. An intelligent method is required to safeguard users from cyber-attacks. In this study, the author put forth a machine learning-based URL detection technique. To identify phishing URLs, a recurrent neural network technique is used. With 7900 malicious and 5800 legitimate sites, researchers tested the suggested method. Website phishing is a type of cyberattack that aims to steal sensitive data from online users, such as login credentials and banking information. Attackers deceive users into thinking that a hidden webpage they are using to access their crucial information is legitimate or reliable. Heuristics, blacklisting or whitelisting, and Machine Learning based techniques are just a few of the solutions to phishing website attacks that have been suggested. The most recent methods for detecting phishing websites using ML techniques are presented in this paper. Based on ML techniques, this research suggests fixes for the website's phishing issue. Challenges to ML techniques identified in this work include overfitting, low accuracy, and ML techniques' ineffectiveness in case of unavailability of enough training data.

INTRODUCTION :-

Phishing is a fraudulent method that obtains customer identification information and financial credentials through the use of technological and social cunning. Social media platforms use spoof emails from real businesses and organizations to give users access to bogus websites where they can reveal financial information like usernames and passwords . Hackers often use tools to intercept usernames and passwords from users' online accounts when they want to steal credentials from computers.

Phishers can steal user information using a variety of channels, such as email, Uniform Resource Locators , instant messages, forum postings, phone calls, and text messages. Phishing content impersonates legitimate content in structure and tempts users to access it in order to obtain their sensitive data. Phishing's main goal is to obtain specific personal information for financial gain or identity theft. Phishing attacks are seriously harming the global economy. Additionally, the majority of phishing attacks target webmail and financial/payment institutions, according to the Anti-Phishing Working Group's most recent studies on phishing patterns.

Criminals create unauthorized copies of legitimate websites and emails, typically from financial institutions or other organizations that deal with financial data, in order to receive confidential information . The logos and taglines of an authentic company were used to create this email. Images or an entire website can be copied thanks to HTML's design and structure . Additionally, it contributes to the Internet's explosive growth as a communication tool and makes it possible for customers to misuse company names, logos, and other identifiers that serve as authentication mechanisms . Phisher distributes "spooled" emails to a large number of recipients in order to trick users. When these emails are opened, customers frequently get redirected away from the real business to a fake website.

There is a good chance that user information will be misused. Because of these factors, phishing is a very urgent, difficult, and important issue in contemporary society . The characteristics of a domain, such as website URLs, website content, incorporating both the website URLs and content, the website source code, and the website screenshot, have been the subject of several recent studies against phishing . To safeguard its users, an organization needs effective anti-phishing tools that can identify malicious URLs.

The traditional method of URL detection relies on a blacklist (a collection of malicious URLs) compiled through user reports or expert judgment. The blacklist is used to verify URLs on the one hand, and the URLs on the blacklist are regularly updated on the other. However, the amount of malicious URLs that are not on the blacklist is significantly rising. For instance, cybercriminals can make new malicious URLs by using a Domain Generation Algorithm (DGA) to get around the blacklist. As a result, it is nearly impossible to identify the malicious URLs without an exhaustive blacklist of malicious URLs . Thus, the current methods cannot detect new malicious URLs.

One of the machine learning (ML) techniques that offers a solution for the challenging real-time problems is the Recurrent Neural Network (RNN)—Long Short-Term Memory (LSTM) . RNN can store inputs for longer with the help of LSTM . It is comparable to the idea of computer storage. Every feature will also be processed using a uniform distribution . A significant amount of information can be extracted from a small number of data points using RNN and LSTM together. In order to quickly identify a malicious website, it supports phishing detection systems.

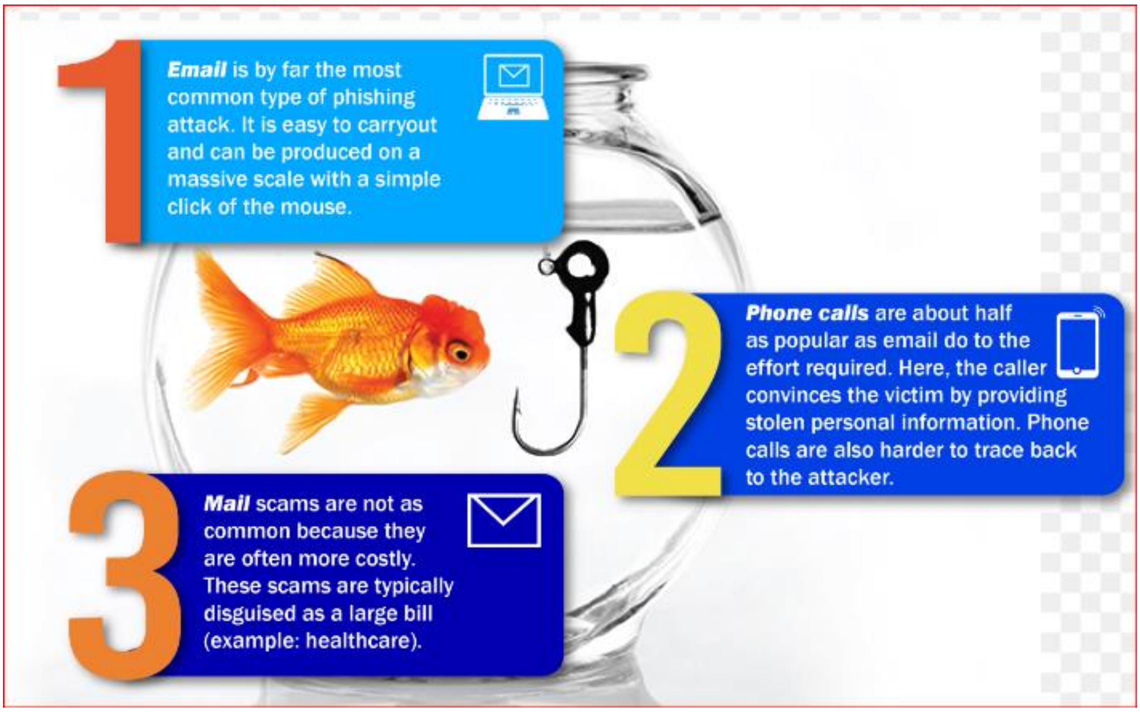
In contrast to the majority of earlier methods, researchers concentrate on picking out malicious URLs from the enormous collection of URLs. As a result, the study suggests a URL detection method based on recurrent neural networks (RNN). The following are the study's goals:

1. To develop a novel approach to detect malicious URL and alert users.
2. To apply ML techniques in the proposed approach in order to analyze the real time URLs and produce effective results.
3. To implement the concept of RNN, which is a familiar ML technique that has the capability to handle huge amount of data.

Research background and related works :

Phishing attacks are categorized according to Phisher’s mechanism for trapping alleged users. Several forms of these attacks are keyloggers, DNS toxicity, Etc. The initiation processes in social engineering include online blogs, short message services , social media platforms that use web 2.0 services, such as Facebook and Twitter, file-sharing services for peers, Voice over IP (VoIP) systems where the attackers use caller spoofing IDs . Each form of phishing has a little difference in how the process is carried out in order to defraud the unsuspecting consumer. Email phishing attacks occur when an attacker sends an e-mail with a link to potential users to direct them to phishing websites.

What is Phishing : -



1 **Email** is by far the most common type of phishing attack. It is easy to carryout and can be produced on a massive scale with a simple click of the mouse.

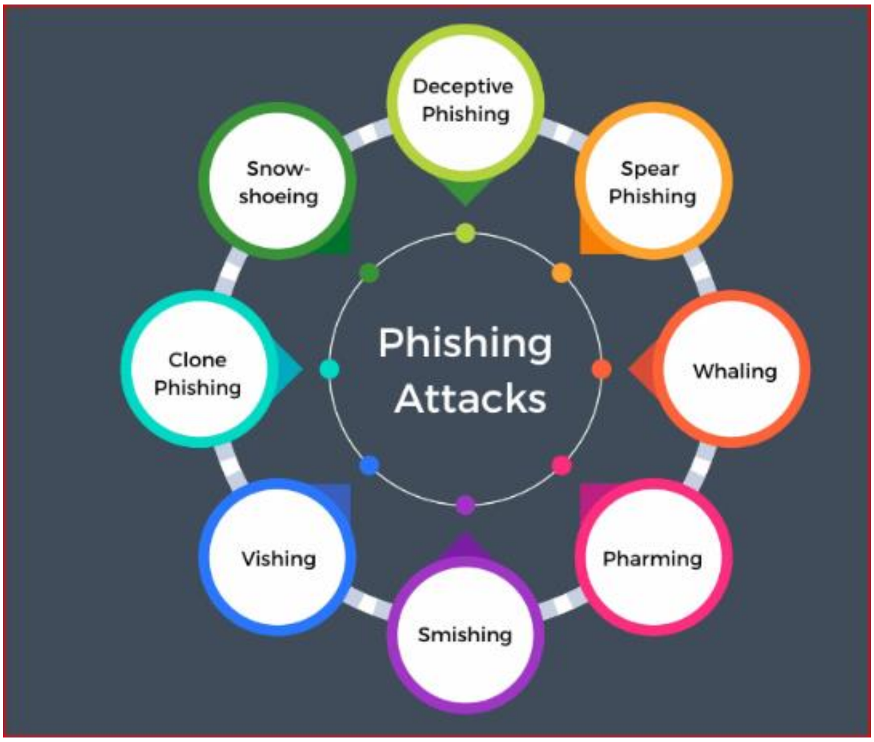
2 **Phone calls** are about half as popular as email do to the effort required. Here, the caller convinces the victim by providing stolen personal information. Phone calls are also harder to trace back to the attacker.

3 **Mail** scams are not as common because they are often more costly. These scams are typically disguised as a large bill (example: healthcare).

Phishing is a type of cybersecurity attack that attempts to obtain data that are sensitive like Username, Password, and more. It attacks the user through mail, text, or direct messages. Now the attachment sends by the attacker is opened by the user because the user thinks that the email, mail, phone calls messages came from a trusted source. It is a type of Social Engineering Attack. For Example, The user may find some messages like the lottery winner. When the user clicks on the attachment the malicious code activates that can access sensitive information details. Or if the user clicks on the link that was sent in the attachment they may be redirected to a different website that will ask for the login credentials of the bank.

Types OF PHISHING ATTACTS :-

There are mainly 7 – types of Phishing Attacts :



1. **Spear Phishing –**

   This attack is used to target any specific organization or an individual for unauthorized access. These types of attacks are not initiated by any random hacker, but these attacks are initiated by someone who seeks information related to financial gain or some important information. Just like the phishing attack spear-phishing also comes from a trusted source. This type of attack is much successful. It is considered to be one of the most successful methods as both of the attacks(that is phishing and spear-phishing) is an online attack on users.

2. **Clone Phishing –**

   This attack is actually based on copying the email messages that were sent from a trusted source. Now the hackers alter the information by adding a link that redirects the user to a malicious or fake website. Now, this is sent to a large number of users and the person who initiated it watches who clicks on the attachment that was sent as a mail. This spreads through the contacts of the user who has clicked on the attachment.

3. **Catphishing –**
   It is a type of social engineering attack that plays with the emotions of a person and exploits them to gain money and information. They target them through dating sites. It is a type of engineering threat.

4. **Voice Phishing –**
   Some attacks require to direct the user through fake websites, but some attacks do not require a fake website. This type of attack is sometimes referred to as vishing. Someone who is using the method of vishing, use modern caller id spoofing to convince the victim that the call is from a trusted source. They also use IVR to make it difficult for the legal authorities to trace, block, monitor. It is used to steal credit card numbers or some confidential data of the user. This type of phishing can cause more harm.

5. **SMS phishing –**
   These attacks are used to make the user revealing account information. This attack is also similar to the phishing attack used by cybercriminals to steal credit card details or sensitive information, by making it look like it came from a trusted organization. Cybercriminals use text messages to get personal information by trying to redirect them to a fake website. This fake website looks like that it is an original website.

6. **Whaling –**

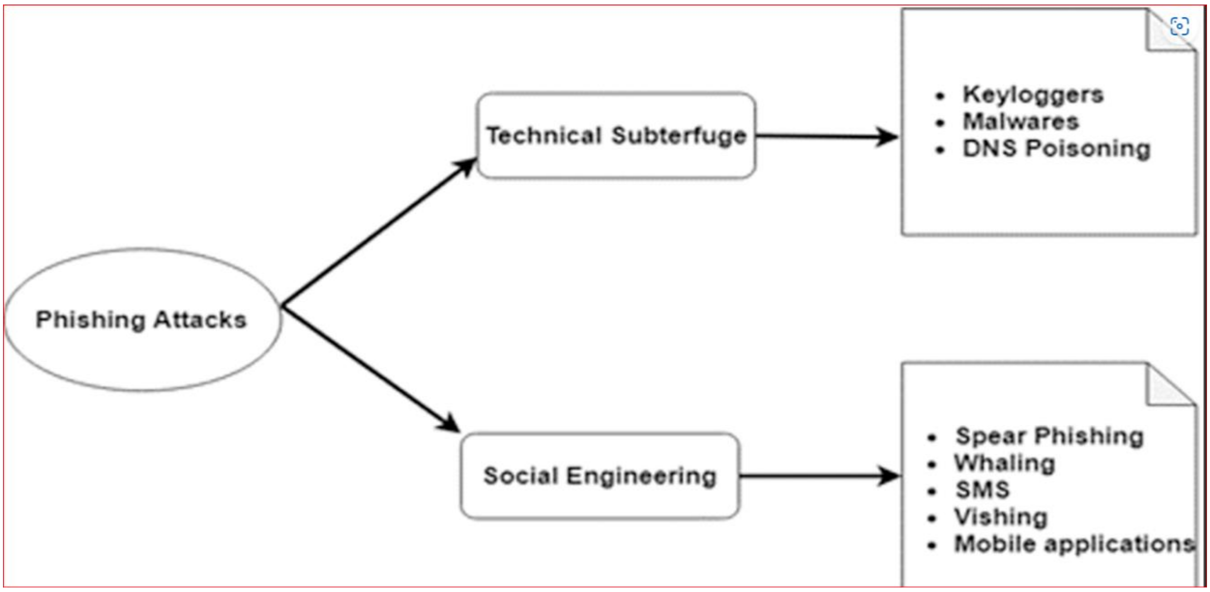
   A whaling phishing attack is defined as a cyber attack wherein cybercriminals disguise themselves as members of a senior management team or other high-power executives of an establishment to target individuals within the organization, either to siphon off money or access sensitive information for malicious purposes. This article covers the basics of whaling phishing, ways to identify an attack, and lists some best practices to prevent them.

7.Decptive Phishing –

Deceptive phishing is the **most common type of phishing**. In this case, an attacker attempts to obtain confidential information from the victims. Attackers use the information to steal money or to launch other attacks. A fake email from a bank asking you to click a link and verify your account details is an example of deceptive phishing.

Classification of phishing attack techniques :

Due to their similarities to legitimate websites, phishing websites present difficulties for businesses and individuals . The different types of phishing attacks are shown in Fig. 1. Keylogging, DNS poisoning, and malware attacks are examples of technical subterfuge. The attacker in these attacks seeks to gain access using a tool or technique. Users trust the network, but the network has been infiltrated by the attackers on the other hand. Spear phishing, Whaling, SMS, Vishing, and mobile applications are examples of social engineering attacks. Attackers target a specific group of people or business in these attacks and con them into visiting a phishing URL . In addition to these attacks, the number of new attacks is growing exponentially as technology advances continuously.



LITERATURE SURVEY :-

In emerging technology, industry, which deeply influence today’s security problems, has given a headache to many employers and home users.

Occurrences that exploit human vulnerabilities have been on the upsurge in recent years. In these new times there are many security systems being enabled to ensure security is given the outmost priority and prevention to be taken from being hacked by those who are involved in cyber-offenses and essential prevention is taken as high importance in organization to ensure network security is not being compromised. Cyber security employee are currently searching for trustworthy and steady detection techniques for phishing websites detection.

Due to wide usage of internet to perform various activities such as online bill payment, banking transaction, online shopping, etc. Customer face numerous security threats like cybercrime. Many cybercrime is being casually executed for example spam, fraud, identity theft cyber terrorisms and phishing. Among this phishing is known as the most common cybercrime today. Phishing has become one amongst the top three most current methods of law breaking in line with recent reports, and both frequency of events and user weakness has increased in recent years, more combination of all these methods result in greater danger of economic damage.

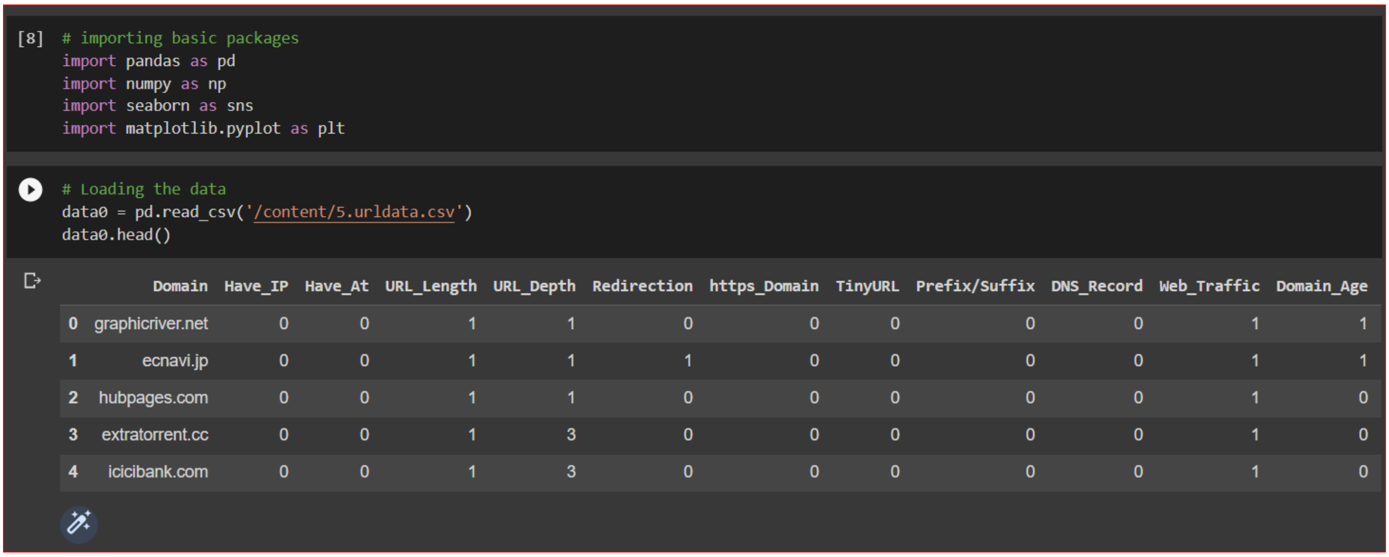
Phishing is a social engineering attack that targets and exploiting the weakness found in the system at the user's end. This paper proposes the Agile Unified Process (AUP) to detect duplicate websites that can potentially collect sensitive information about the user. The system checks the blacklisted sites in dataset and learns the patterns followed by the phishing websites and applies it to further given inputs. The system sends a pop-up and an e-mail notification to the user, if the user clicks on a phishing link and redirects to the site if it is a safe website. This system does not support real time detection of phishing sites; user has to supply the website link to the system developed with Microsoft Visual Studio 2010 Ultimate and MySQL stocks up data and to implement database in this system.

1. *LOARDING DATA* :-

The features are extracted and store in the csv file. The working of this can be seen in the 'Phishing Website Feature Extraction.ipynb' file.

The reulted csv file is uploaded to this notebook and stored in the dataframe.

Importing basic packages :



```
[8] # importing basic packages
    import pandas as pd
    import numpy as np
    import seaborn as sns
    import matplotlib.pyplot as plt
```

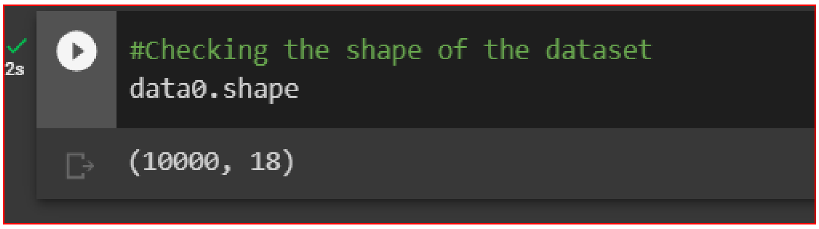
```
# Loading the data
data0 = pd.read_csv('/content/5.urldata.csv')
data0.head()
```

	Domain	Have_IP	Have_At	URL_Length	URL_Depth	Redirection	https_Domain	TinyURL	Prefix/Suffix	DNS_Record	Web_Traffic	Domain_Age
0	graphicriver.net	0	0	1	1	0	0	0	0	0	1	1
1	ecnavi.jp	0	0	1	1	1	0	0	0	0	1	1
2	hubpages.com	0	0	1	1	0	0	0	0	0	1	0
3	extratorrent.cc	0	0	1	3	0	0	0	0	0	1	0
4	icicibank.com	0	0	1	3	0	0	0	0	0	1	0

2. Familiarizing With Data :-

In this step of familiarizing with dataset, few dataframe methods are used to look into the data and its features.

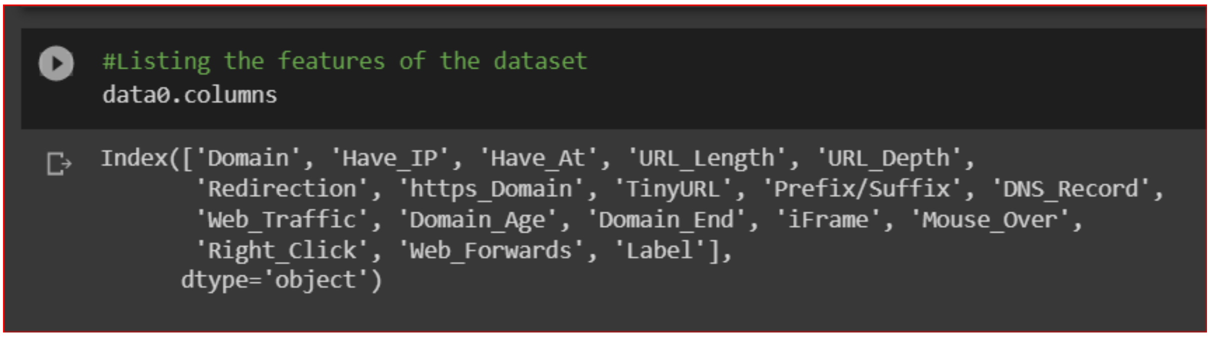
2.1) Checking the shape of the Dataset.



```
#Checking the shape of the dataset
data0.shape
```

```
(10000, 18)
```

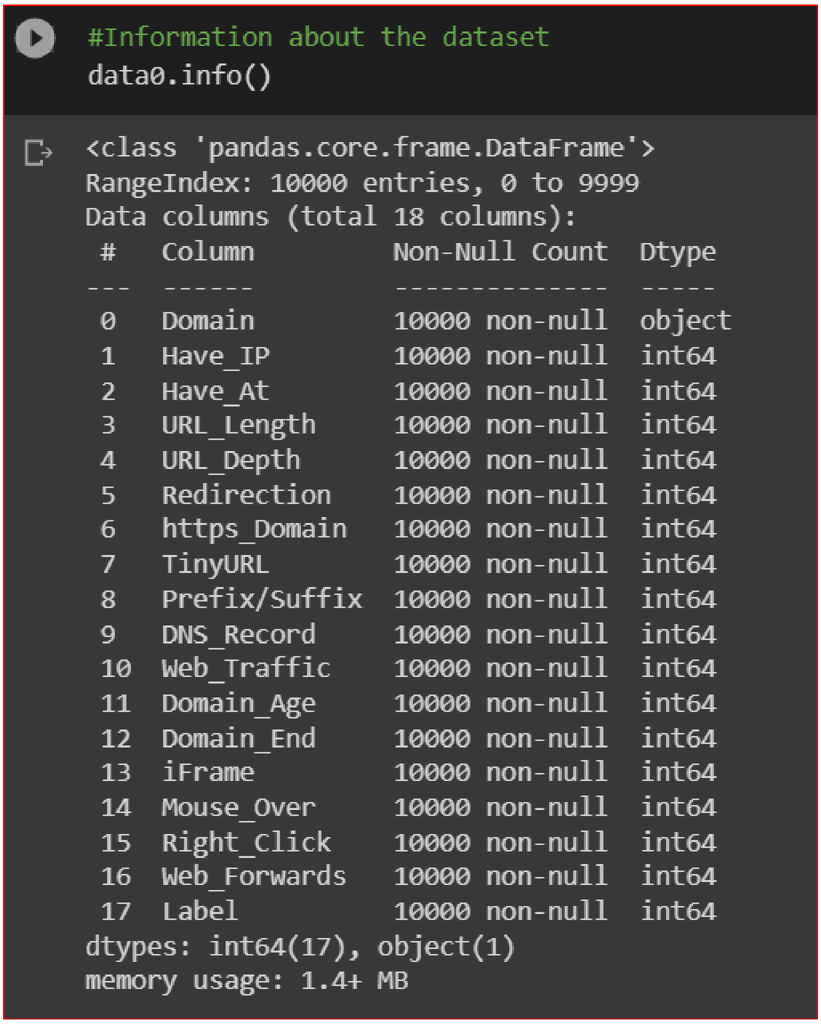
2.2) Listing the Features of the Dataset.



```
#Listing the features of the dataset
data0.columns
```

```
Index(['Domain', 'Have_IP', 'Have_At', 'URL_Length', 'URL_Depth',
       'Redirection', 'https_Domain', 'TinyURL', 'Prefix/Suffix', 'DNS_Record',
       'Web_Traffic', 'Domain_Age', 'Domain_End', 'iFrame', 'Mouse_Over',
       'Right_Click', 'Web_Forwards', 'Label'],
      dtype='object')
```

2.3) Information about the dataset.

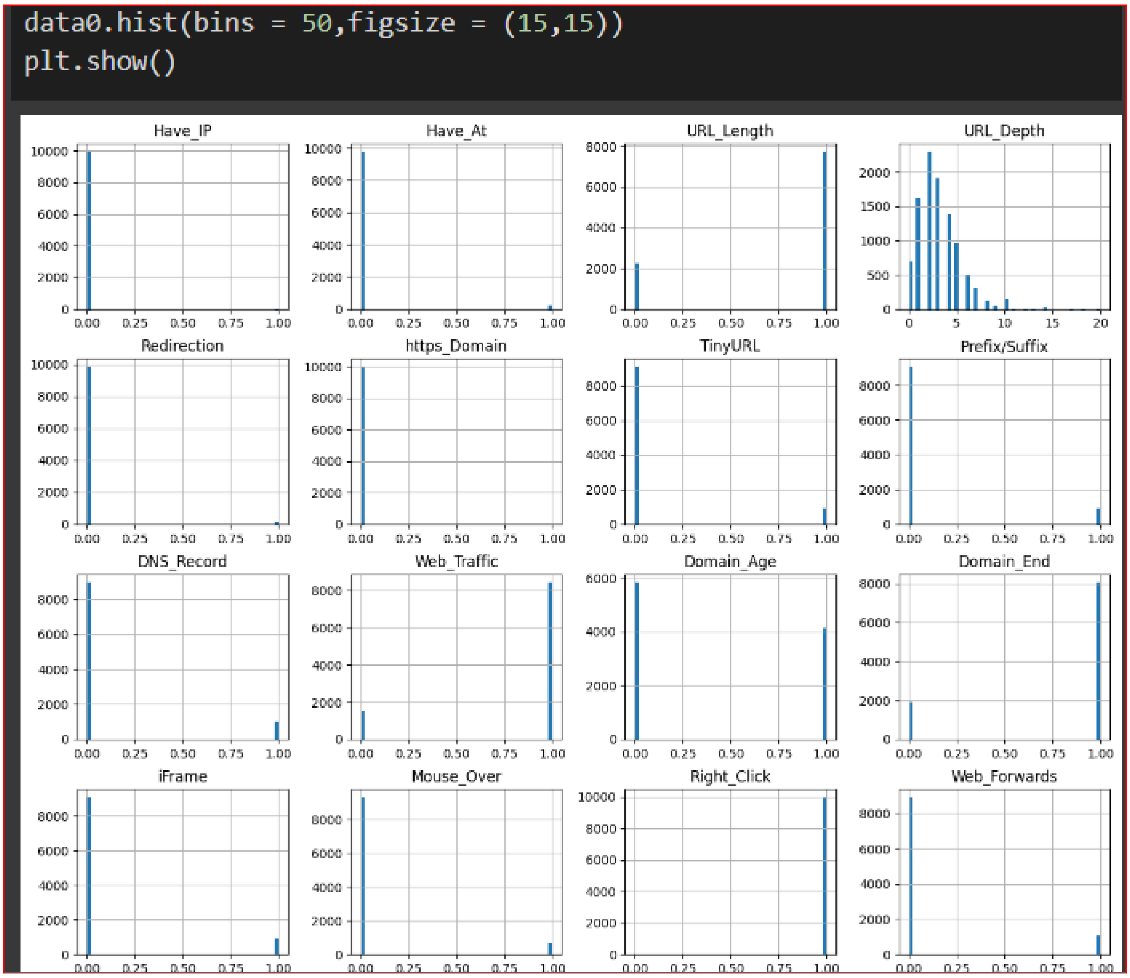


```
#Information about the dataset
data0.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 18 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Domain         10000 non-null  object
 1   Have_IP        10000 non-null  int64
 2   Have_At        10000 non-null  int64
 3   URL_Length     10000 non-null  int64
 4   URL_Depth      10000 non-null  int64
 5   Redirection    10000 non-null  int64
 6   https_Domain   10000 non-null  int64
 7   TinyURL        10000 non-null  int64
 8   Prefix/Suffix  10000 non-null  int64
 9   DNS_Record     10000 non-null  int64
 10  Web_Traffic    10000 non-null  int64
 11  Domain_Age     10000 non-null  int64
 12  Domain_End     10000 non-null  int64
 13  iFrame         10000 non-null  int64
 14  Mouse_Over     10000 non-null  int64
 15  Right_Click    10000 non-null  int64
 16  Web_Forwards   10000 non-null  int64
 17  Label          10000 non-null  int64
dtypes: int64(17), object(1)
memory usage: 1.4+ MB
```

3) Visualizing the Data: -

To determine how the data is distributed and how features relate to one another, a few plots and graphs are displayed.

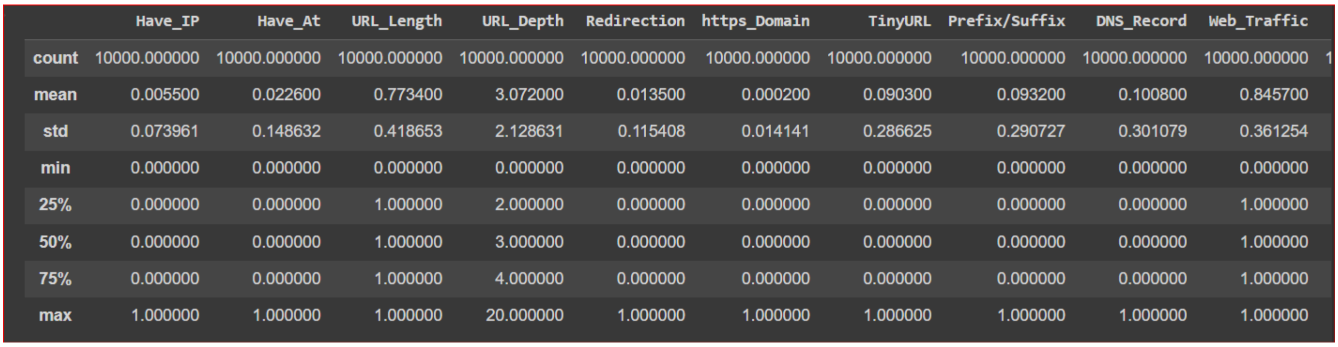


```
data0.hist(bins = 50,figsize = (15,15))
plt.show()
```

4) Data Preprocessing & EDA

Here, we clean the data by applying data preprocesssing techniques and transform the data to use it in the models.

[ ] data0.describe()



	Have_IP	Have_At	URL_Length	URL_Depth	Redirection	https_Domain	TinyURL	Prefix/Suffix	DNS_Record	Web_Traffic
count	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000
mean	0.005500	0.022600	0.773400	3.072000	0.013500	0.000200	0.090300	0.093200	0.100800	0.845700
std	0.073961	0.148632	0.418653	2.128631	0.115408	0.014141	0.286625	0.290727	0.301079	0.361254
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	1.000000	2.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000
50%	0.000000	0.000000	1.000000	3.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000
75%	0.000000	0.000000	1.000000	4.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000
max	1.000000	1.000000	1.000000	20.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000

The above obtains result shows that the most of the data made O’s & 1’s except ‘Domains’ & ‘URL_Depth’ columns. The Domins Column doesn’t have any singficance to the machine learing models training. So dropping the ‘Domins’ column from the dataset.

4.1) **Dropping the Domain column**

```
data = data0.drop(['Domain'], axis = 1).copy()
```

- This leaves us with 16 features & a target column. The 'URL_Depth' maximum value is 20. According to my understanding, there is no necessity to change this column.

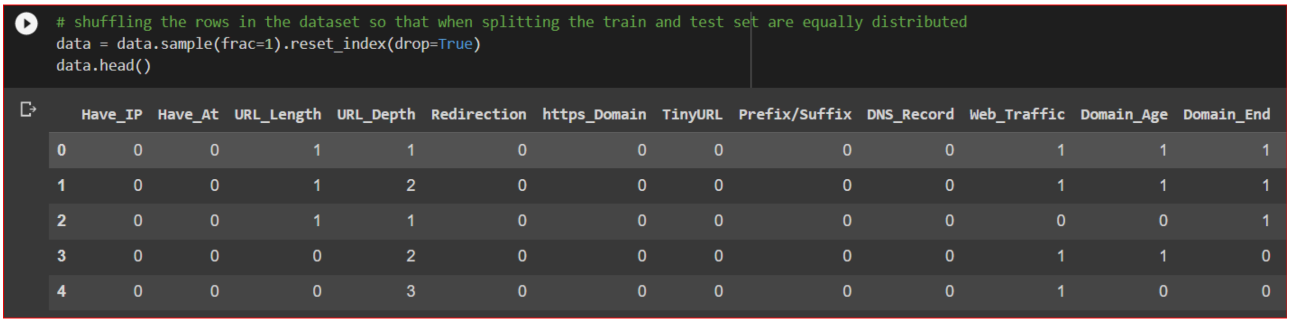
4.2) **checking the data for null or missing values**

```
data.isnull().sum()
```

- In the feature extraction file, the extracted features of legitmate & phishing url datasets are just concatenated without any shuffling. This resulted in top 5000 rows of legitimate url data & bottom 5000 of phishing url data.

To even out the distribution while splitting the data into training & testing sets, we need to shuffle it. This even evades the case of overfitting while model training.

4.3) **Shuffting the rows in the dataset so that splitting thr train and test set are equally distributed.**



```
# shuffling the rows in the dataset so that when splitting the train and test set are equally distributed
data = data.sample(frac=1).reset_index(drop=True)
data.head()
```

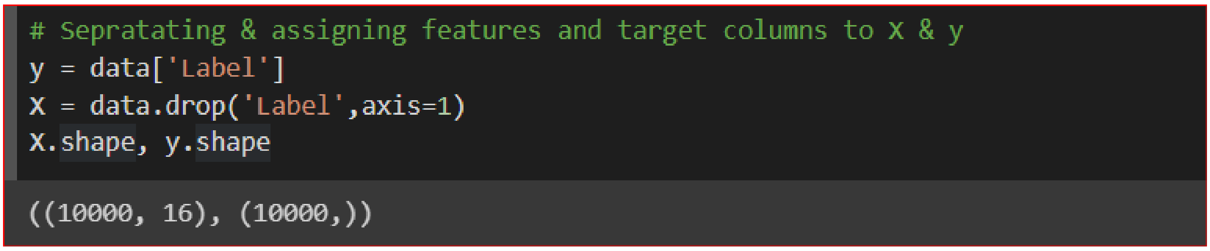
	Have_IP	Have_At	URL_Length	URL_Depth	Redirection	https_Domain	TinyURL	Prefix/Suffix	DNS_Record	Web_Traffic	Domain_Age	Domain_End
0	0	0	1	1	0	0	0	0	0	1	1	1
1	0	0	1	2	0	0	0	0	0	1	1	1
2	0	0	1	1	0	0	0	0	0	0	0	1
3	0	0	0	2	0	0	0	0	0	1	1	0
4	0	0	0	3	0	0	0	0	0	1	0	0

From the above execution, it is clear that the data doesnot have any missing values.

So that, the data is throughly preprocessed & is ready for training.

5) Splitting the data

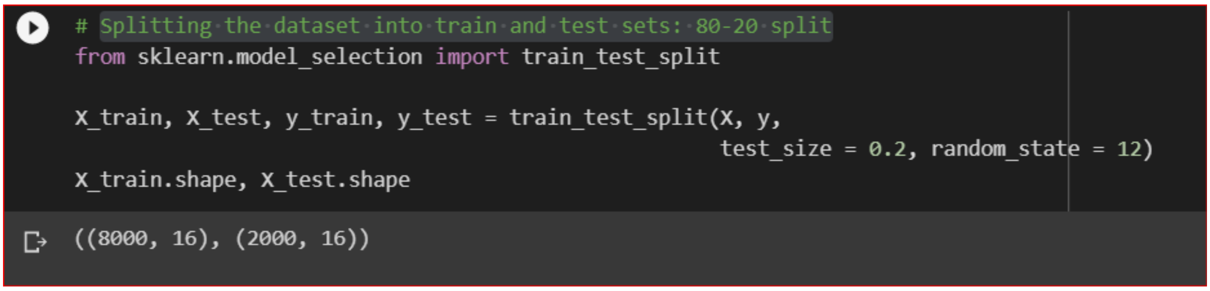
5.1)Sepratating & assigning features and target columns to X & y.



```
# Sepratating & assigning features and target columns to X & y
y = data['Label']
X = data.drop('Label',axis=1)
X.shape, y.shape
```

```
((10000, 16), (10000,))
```

5.2) Splitting the dataset into train and test sets: 80-20 split



```
# Splitting the dataset into train and test sets: 80-20 split
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    test_size = 0.2, random_state = 12)
X_train.shape, X_test.shape
```

```
((8000, 16), (2000, 16))
```

6)Machine Learning Models & Training :

From the dataset above, it is clear that this is a supervised machine learning task. There are two major types of supervised machine learning problems, called classification and regression.

This data set comes under classification problem, as the input URL is classified as phishing (1) or legitimate (0). The supervised machine learning models (classification) considered to train the dataset in this notebook are:

1. Decision Tree.
2. Random Forest.
3. Multilayer Perceptrons.
4. XGBoost.
5. Autoencoder Neural Network.
6. Support Vector Machines.

6.1) **Decision Tree** :-

Decision trees are widely used models for classification and regression tasks. Essentially, they learn a hierarchy of if/else questions, leading to a decision. Learning a decision tree means learning the sequence of if/else questions that gets us to the true answer most quickly.

In the machine learning setting, these questions are called tests (not to be confused with the test set, which is the data we use to test to see how generalizable our model is). To build a tree, the algorithm searches over all possible tests and finds the one that is most informative about the target variable.

```
from sklearn.tree import DecisionTreeClassifier

# instantiate the model
tree = DecisionTreeClassifier(max_depth = 5)
# fit the model
tree.fit(X_train, y_train)
```

Performance Evaluation:

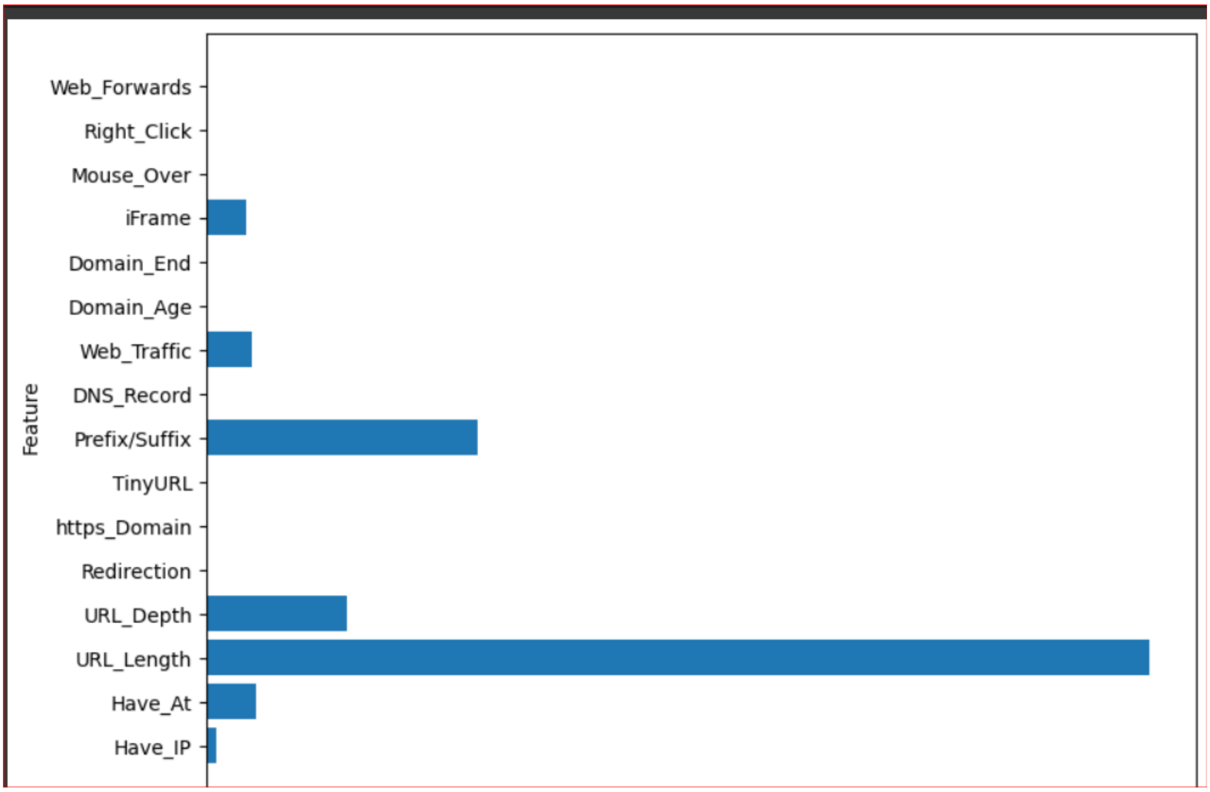
computing the accuracy of the model performance :

```
acc_train_tree = accuracy_score(y_train,y_train_tree)
acc_test_tree = accuracy_score(y_test,y_test_tree)

print("Decision Tree: Accuracy on training Data: {:.3f}".format(acc_tra
in_tree))
print("Decision Tree: Accuracy on test Data: {:.3f}".format(acc_test_tr
ee))
```

checking the feature improtance in the model :

```
plt.figure(figsize=(9,7))
n_features = X_train.shape[1]
plt.barh(range(n_features), tree.feature_importances_, align='center')
plt.yticks(np.arange(n_features), X_train.columns)
plt.xlabel("Feature importance")
plt.ylabel("Feature")
plt.show()
```



6.2)**Random Forest Classifier :-**

Random forests for regression and classification are currently among the most widely used machine learning methods.A random forest is essentially a collection of decision trees, where each tree is slightly different from the others. The idea behind random forests is that each tree might do a relatively good job of predicting, but will likely overfit on part of the data.

If we build many trees, all of which work well and overfit in different ways, we can reduce the amount of overfitting by averaging their results. To build a random forest model, you need to decide on the number of trees to build (the n_estimators parameter of RandomForestRegressor or RandomForestClassifier). They are very powerful, often work well without heavy tuning of the parameters, and don’t require scaling of the date.

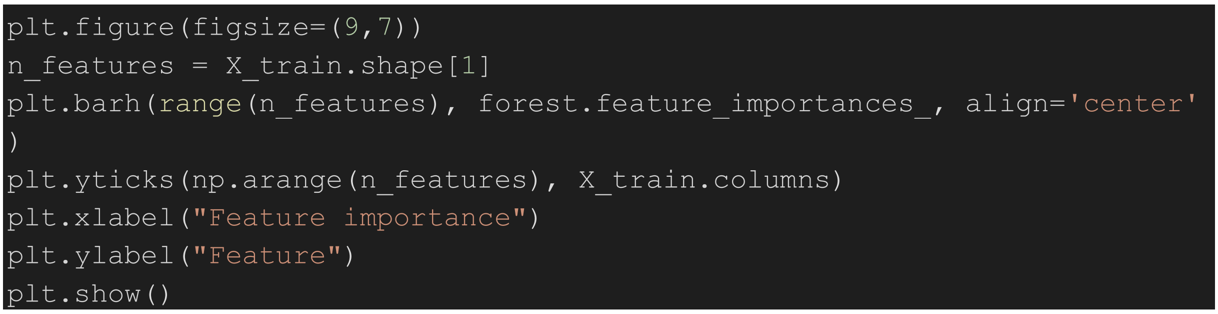
Performance Evaluation:

The accuracy of the model performance :

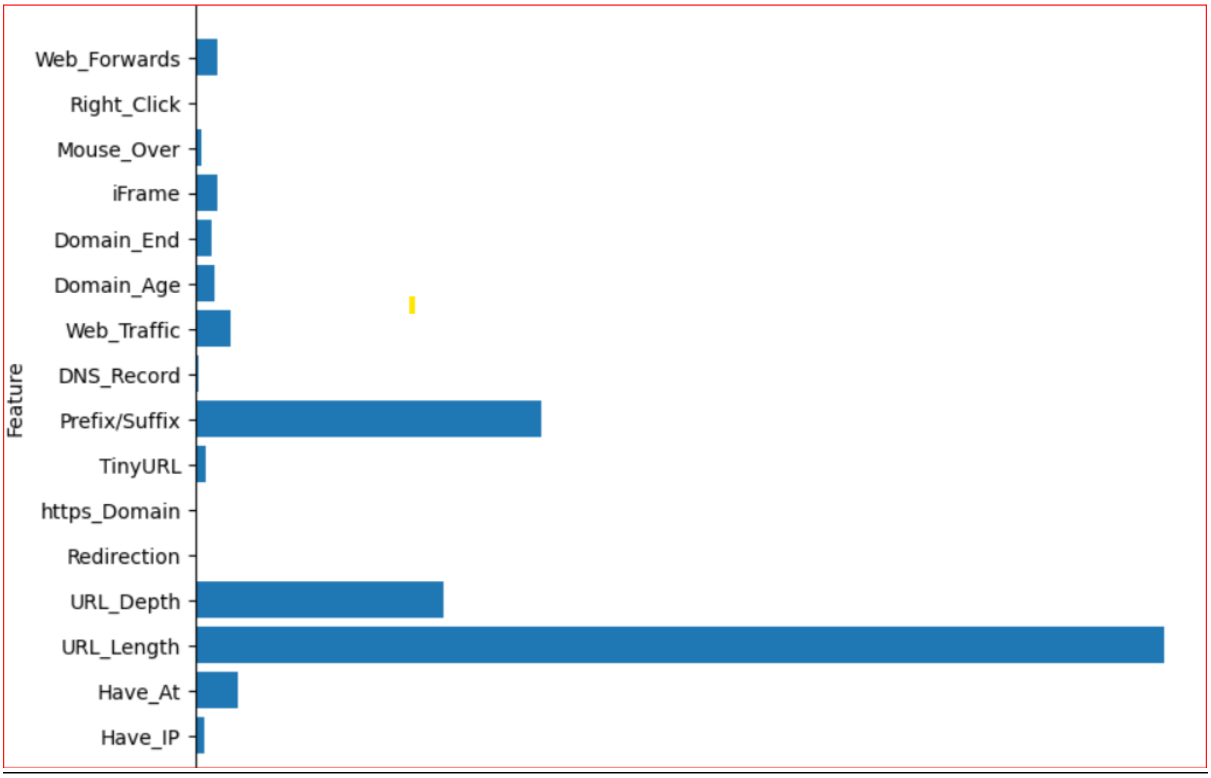
```
acc_train_forest = accuracy_score(y_train,y_train_forest)
acc_test_forest = accuracy_score(y_test,y_test_forest)

print("Random forest: Accuracy on training Data: {:.3f}".format(acc_tra
in_forest))
print("Random forest: Accuracy on test Data: {:.3f}".format(acc_test_fo
rest))
```

Checking the feature improtance in the model :-



```
plt.figure(figsize=(9,7))
n_features = X_train.shape[1]
plt.barh(range(n_features), forest.feature_importances_, align='center'
)
plt.yticks(np.arange(n_features), X_train.columns)
plt.xlabel("Feature importance")
plt.ylabel("Feature")
plt.show()
```



6.3Multilayer Perceptrons (MLPs): Deep Learning :-

Multilayer perceptrons (MLPs) are also known as (vanilla) feed-forward neural networks, or sometimes just neural networks. Multilayer perceptrons can be applied for both classification and regression problems.

MLPs can be viewed as generalizations of linear models that perform multiple stages of processing to come to a decision.

6.4)XGBoost Classifier :-

XGBoost is one of the most popular machine learning algorithms these days. XGBoost stands for eXtreme Gradient Boosting. Regardless of the type of prediction task at hand; regression or classification. XGBoost is an implementation of gradient boosted decision trees designed for speed and performance.

Performance Evaluation:

computing the accuracy of the model performance

```
acc_train_xgb = accuracy_score(y_train,y_train_xgb)
acc_test_xgb = accuracy_score(y_test,y_test_xgb)

print("XGBoost: Accuracy on training Data: {:.3f}".format(acc_train_xgb
))
print("XGBoost : Accuracy on test Data: {:.3f}".format(acc_test_xgb))
```

6.5) Autoencoder Neural Network :-

An auto encoder is a neural network that has the same number of input neurons as it does outputs. The hidden layers of the neural network will have fewer neurons than the input/output neurons. Because there are fewer neurons, the auto-encoder must learn to encode the input to the fewer hidden neurons. The predictors (x) and output (y) are exactly the same in an auto encoder.

6.5) Support Vector Machines

In machine learning, support-vector machines (SVMs, also support-vector networks) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier.

Performance Evaluation:

Computing the accuracy of the model performance :

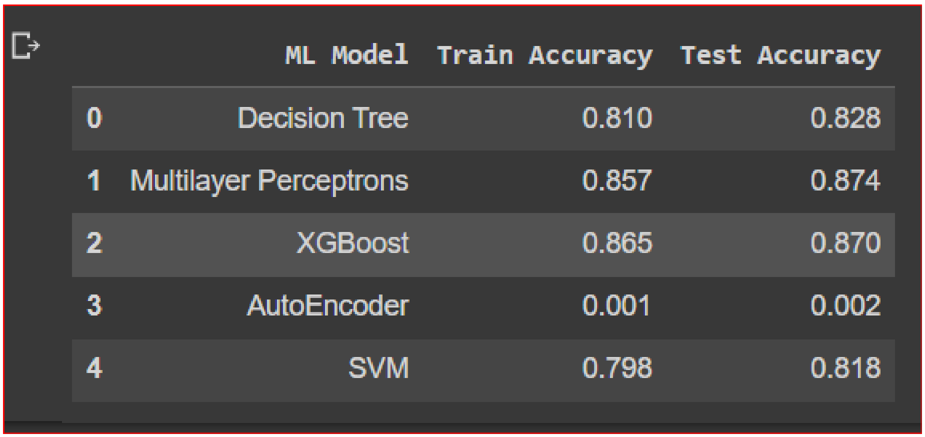
```
acc_train_svm = accuracy_score(y_train,y_train_svm)
acc_test_svm = accuracy_score(y_test,y_test_svm)

print("SVM: Accuracy on training Data: {:.3f}".format(acc_train_svm))
print("SVM : Accuracy on test Data: {:.3f}".format(acc_test_svm))
```

7) Comparision of Models :-

To compare the models performance, a dataframe is created. The columns of thi s dataframe are the lists created to store the results of the model.

```
#creating dataframe
results = pd.DataFrame({ 'ML Model': ML_Model,
    'Train Accuracy': acc_train,
    'Test Accuracy': acc_test})
results
```



	ML Model	Train Accuracy	Test Accuracy
0	Decision Tree	0.810	0.828
1	Multilayer Perceptrons	0.857	0.874
2	XGBoost	0.865	0.870
3	AutoEncoder	0.001	0.002
4	SVM	0.798	0.818

For the above comparision, it is clear that the XGBoost Classifier works well with this dataset.

So, saving the model for future use.