Website Phishing

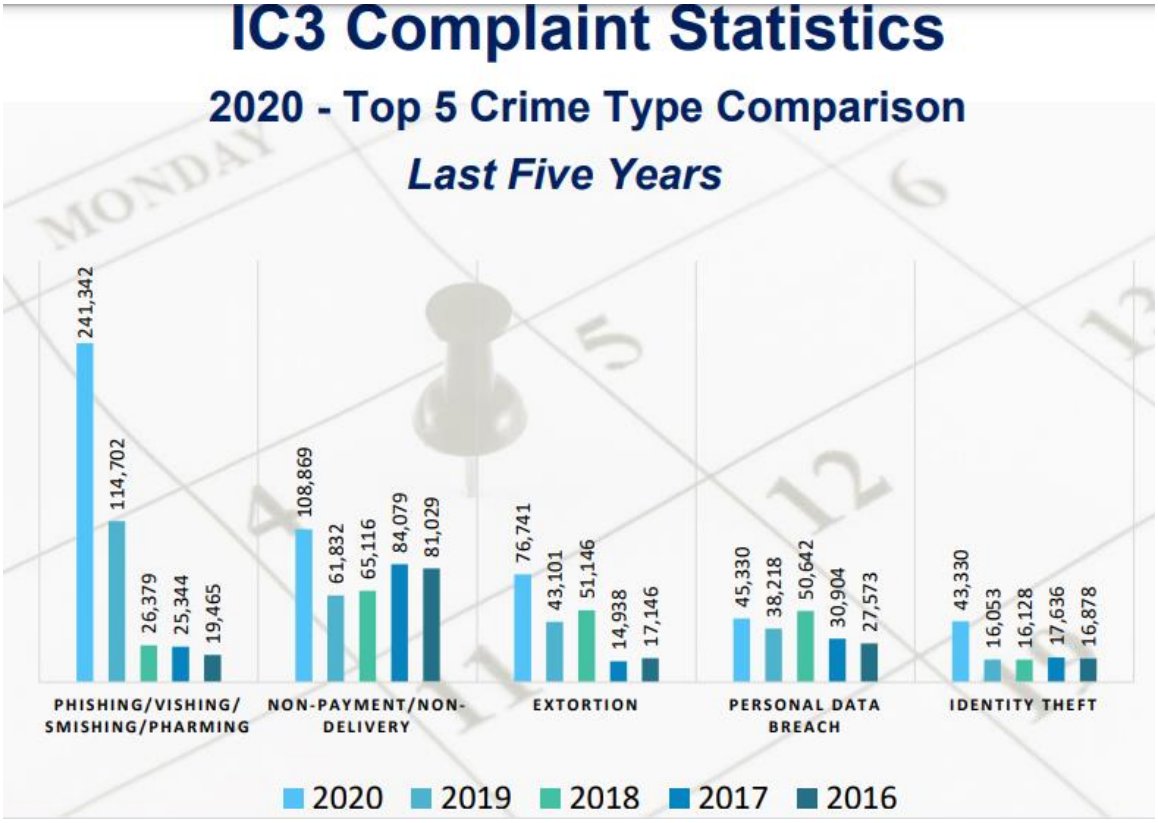
Introduction

Phishing is the fraudulent attempt to obtain sensitive information or data, such as usernames, passwords, credit card numbers, or other sensitive details by impersonating oneself as a trustworthy entity in a digital communication. Typically carried out by email spoofing, instant messaging, and text messaging, phishing often directs users to enter personal information at a fake website which matches the look and feel of the legitimate site.

Phishing is by far the most common attack performed by cyber-criminals, recording over twice as many incidents of phishing than any other type of computer crime.

As per Data by FBI, crime related to phishing increased rapidly.

Comparison for the top five reported crime types of 2020 for the years of 2016 to 2020:



Websites Phishing using Supervised Machine Learning

The main idea of this project is to how website phishing attacks can be prevented using supervised machine learning technique. In this project I am going to train the Logistic Regression model with the UCI Phishing Websites Data Set and see how accurate our model differentiate between Phishing and Legitimate website.

In this project I am going to use UCI Phishing Websites Data Set, which contains 11055 instances and 31 attributes.

First, we will import the dataset and see the dataset.

```
from scipy.io import arff
import pandas as pd
import numpy as np
```

```
df=pd.read_csv('D:/Security/Project/Phishing Dataset/csv_result-Training Dataset.csv')
```

```
df.head().T
```

	0	1	2	3	4
having_IP_Address	-1	1	1	1	1
URL_Length	1	1	0	0	0
Shortining_Service	1	1	1	1	-1
having_At_Symbol	1	1	1	1	1
double_slash_redirecting	-1	1	1	1	1
Prefix_Suffix	-1	-1	-1	-1	-1
having_Sub_Domain	-1	0	-1	-1	1
SSLfinal_State	-1	1	-1	-1	1
Domain_registeration_length	-1	-1	-1	1	-1
Favicon	1	1	1	1	1
port	1	1	1	1	1
HTTPS_token	-1	-1	-1	-1	1
Request_URL	1	1	1	-1	1
URL_of_Anchor	-1	0	0	0	0
Links_in_tags	1	-1	-1	0	0
SFH	-1	-1	-1	-1	-1
Submitting_to_email	-1	1	-1	1	1
Abnormal_URL	-1	1	-1	1	1
Redirect	0	0	0	0	0
on_mouseover	1	1	1	1	-1
RightClick	1	1	1	1	1
popUpWidnow	1	1	1	1	-1
Iframe	1	1	1	1	1
age_of_domain	-1	-1	1	-1	-1
DNSRecord	-1	-1	-1	-1	-1
web_traffic	-1	0	1	1	0
Page_Rank	-1	-1	-1	-1	-1
Google_Index	1	1	1	1	1
Links_pointing_to_page	1	1	0	-1	1
Statistical_report	-1	1	-1	1	1
Result	-1	-1	-1	-1	1

Checking all columns in the dataset:

```
df.columns
```

```
Index(['having_IP_Address', 'URL_Length', 'Shortining_Service',
       'having_At_Symbol', 'double_slash_redirecting', 'Prefix_Suffix',
       'having_Sub_Domain', 'SSLfinal_State', 'Domain_registeration_length',
       'Favicon', 'port', 'HTTPS_token', 'Request_URL', 'URL_of_Anchor',
       'Links_in_tags', 'SFH', 'Submitting_to_email', 'Abnormal_URL',
       'Redirect', 'on_mouseover', 'RightClick', 'popUpWidnow', 'Iframe',
       'age_of_domain', 'DNSRecord', 'web_traffic', 'Page_Rank',
       'Google_Index', 'Links_pointing_to_page', 'Statistical_report',
       'Class'],
      dtype='object')
```

Now we will check the total number of observations and classes in the dataset:

```
from collections import Counter
```

```
classes = Counter(df['Result'].values)
classes.most_common()
```

```
[(1, 6157), (-1, 4898)]
```

```
class_distribution = pd.DataFrame(classes.most_common(), columns=['Class', 'Number of Observations'])
class_distribution
```

	Class	Number of Observations
0	1	6157
1	-1	4898

As result dataset contains 4898 observation from -1 class shows Phishing data and 6157 observations from 1 class shows Legitimate data.

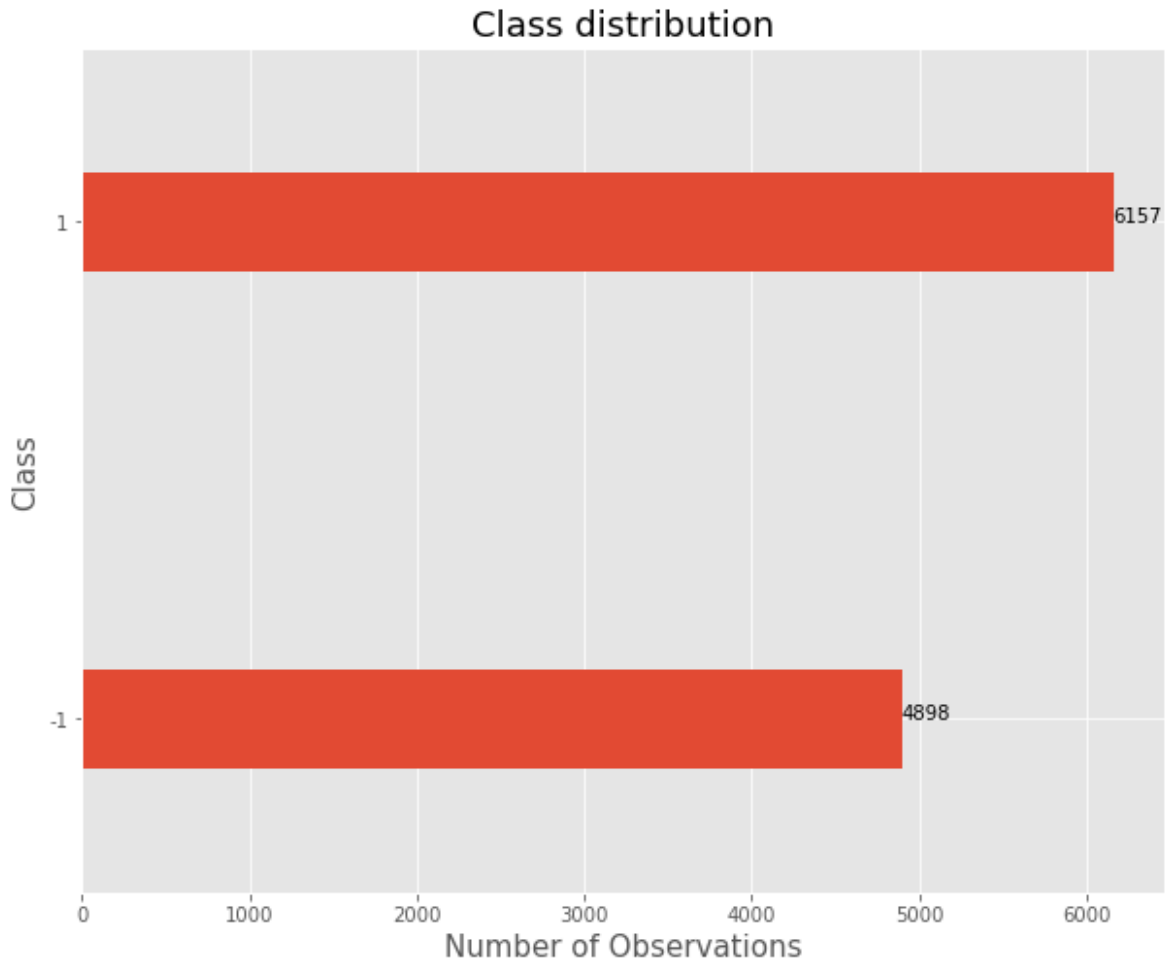
Plotting Class vs Number of Observations:

```
import matplotlib.pyplot as plt
%matplotlib inline
plt.style.use('ggplot')

plot = class_distribution.groupby('Class')['Number of Observations'].sum().plot(kind='barh', width=0.2, figsize=(10,8))

plot.set_title('Class distribution', fontsize = 18)
plot.set_xlabel('Number of Observations', fontsize = 15)
plot.set_ylabel('Class', fontsize = 15)

for i in plot.patches:
    plot.text(i.get_width()+0.1, i.get_y()+0.1,str(i.get_width()), fontsize=10)
```


Class distribution
1
-1
Class
6157
4898
0
1000
2000
3000
4000
5000
6000
Number of Observations

Descriptive Statistics Summary of the dataset:

	count	mean	std	min	25%	50%	75%	max
having_IP_Address	11055.0	0.313795	0.949534	-1.0	-1.0	1.0	1.0	1.0
URL_Length	11055.0	-0.633198	0.766095	-1.0	-1.0	-1.0	-1.0	1.0
Shortining_Service	11055.0	0.738761	0.673998	-1.0	1.0	1.0	1.0	1.0
having_At_Symbol	11055.0	0.700588	0.713598	-1.0	1.0	1.0	1.0	1.0
double_slash_redirecting	11055.0	0.741474	0.671011	-1.0	1.0	1.0	1.0	1.0
Prefix_Suffix	11055.0	-0.734962	0.678139	-1.0	-1.0	-1.0	-1.0	1.0
having_Sub_Domain	11055.0	0.063953	0.817518	-1.0	-1.0	0.0	1.0	1.0
SSLfinal_State	11055.0	0.250927	0.911892	-1.0	-1.0	1.0	1.0	1.0
Domain_registeration_length	11055.0	-0.336771	0.941629	-1.0	-1.0	-1.0	1.0	1.0
Favicon	11055.0	0.628584	0.777777	-1.0	1.0	1.0	1.0	1.0
port	11055.0	0.728268	0.685324	-1.0	1.0	1.0	1.0	1.0
HTTPS_token	11055.0	0.675079	0.737779	-1.0	1.0	1.0	1.0	1.0
Request_URL	11055.0	0.186793	0.982444	-1.0	-1.0	1.0	1.0	1.0
URL_of_Anchor	11055.0	-0.076526	0.715138	-1.0	-1.0	0.0	0.0	1.0
Links_in_tags	11055.0	-0.118137	0.763973	-1.0	-1.0	0.0	0.0	1.0
SFH	11055.0	-0.595749	0.759143	-1.0	-1.0	-1.0	-1.0	1.0
Submitting_to_email	11055.0	0.635640	0.772021	-1.0	1.0	1.0	1.0	1.0
Abnormal_URL	11055.0	0.705292	0.708949	-1.0	1.0	1.0	1.0	1.0
Redirect	11055.0	0.115694	0.319872	0.0	0.0	0.0	0.0	1.0
on_mouseover	11055.0	0.762099	0.647490	-1.0	1.0	1.0	1.0	1.0
RightClick	11055.0	0.913885	0.405991	-1.0	1.0	1.0	1.0	1.0
popUpWidnow	11055.0	0.613388	0.789818	-1.0	1.0	1.0	1.0	1.0
Iframe	11055.0	0.816915	0.576784	-1.0	1.0	1.0	1.0	1.0
age_of_domain	11055.0	0.061239	0.998168	-1.0	-1.0	1.0	1.0	1.0
DNSRecord	11055.0	0.377114	0.926209	-1.0	-1.0	1.0	1.0	1.0
web_traffic	11055.0	0.287291	0.827733	-1.0	0.0	1.0	1.0	1.0
Page_Rank	11055.0	-0.483673	0.875289	-1.0	-1.0	-1.0	1.0	1.0
Google_Index	11055.0	0.721574	0.692369	-1.0	1.0	1.0	1.0	1.0
Links_pointing_to_page	11055.0	0.344007	0.569944	-1.0	0.0	0.0	1.0	1.0
Statistical_report	11055.0	0.719584	0.694437	-1.0	1.0	1.0	1.0	1.0
Result	11055.0	0.113885	0.993539	-1.0	-1.0	1.0	1.0	1.0

Concise summary of the dataset data type:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11055 entries, 0 to 11054
Data columns (total 31 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   having_IP_Address            11055 non-null  int64
 1   URL_Length                   11055 non-null  int64
 2   Shortining_Service           11055 non-null  int64
 3   having_At_Symbol             11055 non-null  int64
 4   double_slash_redirecting     11055 non-null  int64
 5   Prefix_Suffix                11055 non-null  int64
 6   having_Sub_Domain            11055 non-null  int64
 7   SSLfinal_State               11055 non-null  int64
 8   Domain_registeration_length  11055 non-null  int64
 9   Favicon                      11055 non-null  int64
 10  port                         11055 non-null  int64
 11  HTTPS_token                  11055 non-null  int64
 12  Request_URL                  11055 non-null  int64
 13  URL_of_Anchor                11055 non-null  int64
 14  Links_in_tags                11055 non-null  int64
 15  SFH                          11055 non-null  int64
 16  Submitting_to_email          11055 non-null  int64
 17  Abnormal_URL                 11055 non-null  int64
 18  Redirect                     11055 non-null  int64
 19  on_mouseover                 11055 non-null  int64
 20  RightClick                   11055 non-null  int64
 21  popUpWidnow                  11055 non-null  int64
 22  Iframe                       11055 non-null  int64
 23  age_of_domain                11055 non-null  int64
 24  DNSRecord                    11055 non-null  int64
 25  web_traffic                  11055 non-null  int64
 26  Page_Rank                    11055 non-null  int64
 27  Google_Index                 11055 non-null  int64
 28  Links_pointing_to_page       11055 non-null  int64
 29  Statistical_report           11055 non-null  int64
 30  Result                       11055 non-null  int64
dtypes: int64(31)
memory usage: 2.6 MB
```

Renaming "Result" to "Class" and Replacing -1 to 0:

```
df.rename(columns={'Result': 'Class'}, inplace=True)

df['Class'] = df['Class'].map({-1:0, 1:1})
df['Class'].unique()
```

```
array([0, 1], dtype=int64)
```

Checking for NA values:

```
df.isna().sum()
```

```
having_IP_Address              0
URL_Length                     0
Shortining_Service             0
having_At_Symbol               0
double_slash_redirecting       0
Prefix_Suffix                  0
having_Sub_Domain              0
SSLfinal_State                 0
Domain_registeration_length    0
Favicon                        0
port                           0
HTTPS_token                    0
Request_URL                    0
URL_of_Anchor                  0
Links_in_tags                  0
SFH                            0
Submitting_to_email            0
Abnormal_URL                   0
Redirect                       0
on_mouseover                   0
RightClick                     0
popUpWidnow                    0
Iframe                         0
age_of_domain                  0
DNSRecord                      0
web_traffic                    0
Page_Rank                      0
Google_Index                   0
Links_pointing_to_page         0
Statistical_report             0
Class                          0
dtype: int64
```

Now we are going to split the dataset into training and testing set with test size 20% and train size 80% to train the model

```
from sklearn.model_selection import train_test_split
```

```
X = df.iloc[:,0:30].values.astype(int)
y = df.iloc[:,30].values.astype(int)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=np.random.seed(7))
```

Applying the Supervised Machine Learning Algorithm Logistic Regression and fitting the data into model

```
from sklearn.linear_model import LogisticRegression

logistic_Regression = LogisticRegression()
logistic_Regression.fit(X_train, y_train)
```

```
LogisticRegression()
```

Now Predicting and checking accuracy on test data based on Logistic Regression algorithm

```
from sklearn.metrics import accuracy_score, classification_report
score = logistic_Regression.score(X_test, y_test)
print('Accuracy score ',score)
print('\n')
print('Accuracy score of the Logistic Regression Classifier {0:.2f}%'
      .format(accuracy_score(y_test, logistic_Regression.predict(X_test))*100.))
print('\n')
print('*****Classification report of the Logistic Regression classifier*****')
print('\n')
print(classification_report(y_test, logistic_Regression.predict(X_test),
                           target_names=['Phishing Websites', 'Normal Websites']))
```

```
Accuracy score  0.9371325192220714


Accuracy score of the Logistic Regression Classifier 93.71%


*****Classification report of the Logistic Regression classifier*****

                   precision    recall  f1-score   support

Phishing Websites       0.94      0.92      0.93       974
  Normal Websites       0.94      0.95      0.94      1237

         accuracy                           0.94      2211
        macro avg       0.94      0.94      0.94      2211
     weighted avg       0.94      0.94      0.94      2211
```

As above prediction accuracy shows that 93%, which means our model is doing good prediction.

Now we will create the Confusion Matrix:

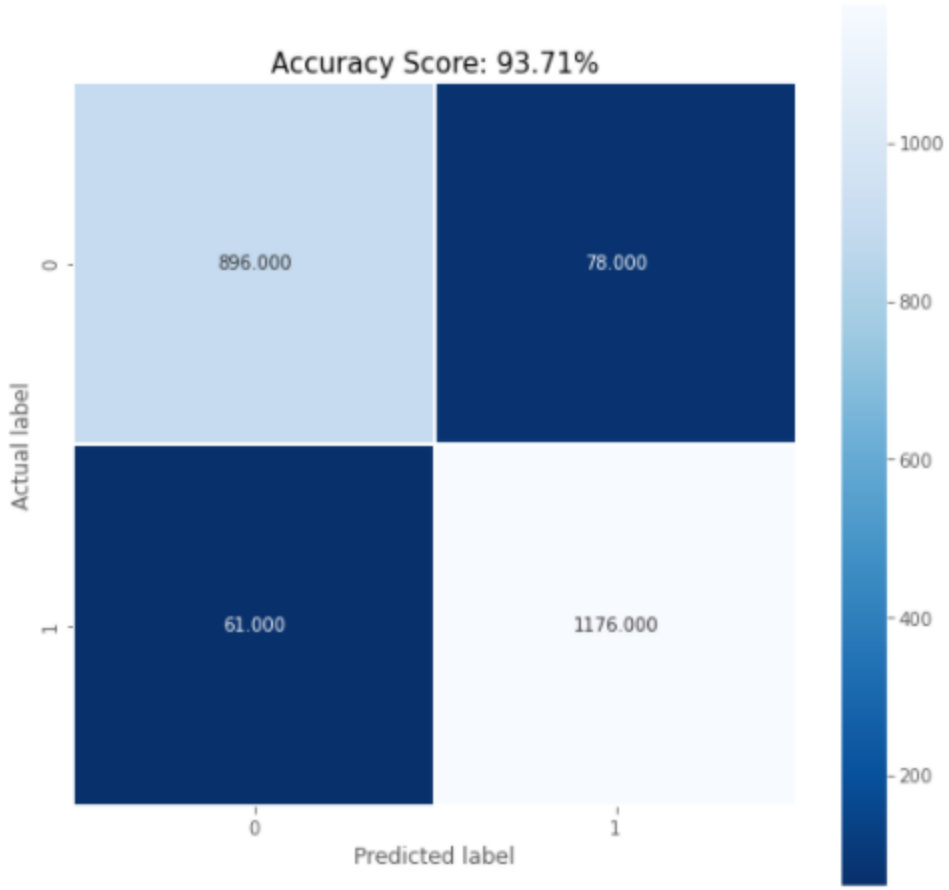
```
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn import metrics
```

```
confusion_matrix = metrics.confusion_matrix(y_test, logistic_Regression.predict(X_test))
print(confusion_matrix)
```

```
[[ 896   78]
 [  61 1176]]
```

Plotting Confusion Matrix:

```
plt.figure(figsize=(9,9))
sns.heatmap(confusion_matrix, annot=True, fmt=".3f", linewidths=.5, square = True, cmap = 'Blues_r');
plt.ylabel('Actual label');
plt.xlabel('Predicted label');
all_sample_title = 'Accuracy Score: {0:.2f}%'.format(accuracy_score(y_test, logistic_Regression.predict(X_test))*100.)
plt.title(all_sample_title, size = 15);
```



The Confusion Matrix tells us the following:

- There are two possible predicted classes: **0** and **1**. If we were predicting that the website is, for example, 0 mean phishing, and 1 means legitimate.
- The classifier made a total of 2211 predictions.
- Out of those 2211 cases, the classifier predicted “0” 896 times, and “1” 1176 times.
- In reality, 1237 data are legitimate and 974 are phishing.

Basic terms related to Confusion matrix

- **True positives (TP):** These are cases in which we predicted one (legitimate), 1176
- **True negatives (TN):** We predicted zero(phishing) ,896
- **False positives (FP):** We predicted one they ***will*** legitimate, but they are not phishing. (Also known as a "Type I error.") 78
- **False negatives (FN):** We predicted zero they are ***not*** legitimate, but they actually phishing (Also known as a "Type II error."), 61

Accuracy: (TP+TN)/Total . Describes overall, how often the classifier correct. i.e. (896+1176)/2211

Comparison with other researchers who have worked on the same dataset:

I learnt from the reference research Paper about assessment of features related to phishing using an automated technique. As per the researcher there are some characteristics that distinguish phishing websites from legitimate ones such as long URL, IP address in URL, adding prefix and suffix to domain and request URL adding prefix and suffix to domain and request URL, etc. In this paper they explore important features that are automatically extracted from websites using a new tool instead of relying on an experienced human in the extraction process and then judge on the features importance in deciding website legitimacy. I used machine learning technique which automatically predict is the website phishing or not considering by using website data to trained the model. Also, my model provide accuracy for the test data.

Summary for Next Step:

In next step I will apply unsupervised machine learning algorithm using K-Means Clustering with same dataset and predict if the website is phishing or not.

References:

1. https://www.fbi.gov/news/pressrel/press-releases/fbi-releases-the-internet-crime-complaint-center-2020-internet-crime-report-including-covid-19-scam-statistics

2. https://archive.ics.uci.edu/ml/datasets/phishing+websites

3. http://www.antiphishing.org/trendsreports/

4. https://archive.ics.uci.edu/ml/machine-learning-databases/00327/Phishing%20Websites%20Features.docx

5.https://www.researchgate.net/publication/261081735_An_assessment_of_features_related_to_phishing_websites_using_an_automated_technique