Phishing Website Detection using Machine Learning Algorithms

A PROJECT REPORT

Submitted by

SAKSHAM DEWAN, 19BCE2090

PALASH SARMA, 19BCE2092

Course Code: CSE3501
Course Title: Information Security Analysis and Audit

Under the guidance of
Dr. Kakelli Anil Kumar
Associate Professor
SCOPE, VIT, Vellore.

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

May -2021

INDEX

ABSTRACT

1. Aim

To detect phishing URLs and categorize the website URLs into ‘good’ and ‘bad’ websites to surf on the internet using machine learning algorithms.

2. Objectives

• To overcome the drawbacks of blacklist and heuristics-based method

• Focus on machine learning techniques.

• Analyze various blacklisted and legitimate URLs and their features to accurately detect the phishing websites including zero- hour phishing websites.

3. Motivation

▪ Phishing becomes a main area of concern for security researchers because it is not difficult to create the fake website which looks so close to legitimate website. Main aim of the attacker is to steal banks account credentials.

▪ Phishing attacks are becoming successful because lack of user awareness.

▪ The general method to detect phishing websites by updating blacklisted URLs, Internet Protocol (IP) to the antivirus database which is also known as “blacklist" method.

▪ To overcome the drawbacks of blacklist and heuristics-based method, many security researchers now focused on machine learning techniques.

▪ Using this technique, algorithm will analyze various blacklisted and legitimate URLs and their features to accurately detect the phishing websites including zero- hour phishing websites.

1. Introduction

Internet plays a critical part in day-to-day activities such as communication, business, transactions, personal needs, marketing, and e-commerce in today's digital world. The internet is a complex tool that allows you to complete a variety of things quickly and easily. In this time of rapid technological advancement, almost everything is now accessible via the internet. As a result, increased internet usage leads to an increase in cybercrime and other virus activities. Online information creates a digital impression, and if it falls into the wrong hands, it may lead to data theft, identity theft, and financial loss. Cybercrime encompasses a wide range of online security threats, with Phishing being one of the most dangerous. Phishing is a deceptive tactic that uses a phishing web page. Phishing uses e-mails and websites that appear to be from a reputable business to dupe customers into divulging personal or financial information. These data are subsequently used by the threatening party for illicit objectives such as identity theft, data theft, and extortion. Clients are tricked into providing sensitive information by filling out a web form, downloading and installing malicious software, or tracking clients' online activities to obtain data. Because of widespread credulity and unawareness, luring Internet users by making them click on rogue links that appear trustworthy is a simple process. It's critical to protect users' personal information from illegal access. The majority of the time, the procedure entails sending messages that direct the recipient to either visit a fraudulent site and enter their data, or to visit an authentic site via a phishing intermediary attack or a spoofed website, which then gathers the user's information and results in several losses. Anti-Phishing strategies are needed to combat the phishing problem. This research proposes a method for identifying and preventing phishing attacks by employing attributes of phishing URLs and a machine learning approach to classify phishing websites.

2. Literature Survey

Authors and Year (Reference)	*Title (Study)*	*Strengths/Important points / Limitations*
Chandrasekaran, M., Narayanan, M. and Upadhyaya, S. (2006)	A Multi-Classifier Based Prediction Model for Phishing Emails Detection Using Topic Modelling, Named Entity Recognition and Image Processing	The model detects phishing by identifying and utilising structural features of email. This classification method's technique is insufficient, and it only employs one approach to identify phishing emails, which is inefficient and scalable. This is solely dependent on e-structural mail's properties, and more structural or content attributes must be added to minimise incorrect results.
Wenyin, Liu & Fang, Ning & Quan, Xiaojun & Qiu, Bite & Liu, Gang. (2010).	Discovering phishing target based on semantic link network	The study presents a novel method for detecting phishing websites by calculating the association relation between webpages, which includes harmful webpages and their connected webpages, in order

		to quantify the combination of link, search, and text relations. To recognise the suspicious webpage as phishing, the semantic link network presents an approach based on four convergent scenarios. The disadvantages of this strategy are that additional types of associations must be made, and visual, layout, and domain commonalities must be linked. This method is considered a time-consuming strategy, and it also requires the investigation of many sub-relations inside the combined association ties.
Almomani, Dr.Ammar & Wan, Tat-Chee & Taha, Altyeb & Manasrah, Ahmad & Almomani, Eman & Anbar, Mohammed & Alomari, Esraa & Ramadass, Sureswaran. (2012).	Evolving Fuzzy Neural Network for Phishing Emails Detection	Phishing email featuring zero-day exploits. In online mode, it distinguishes between phishing and ham email. It is used for feature retrieval, rank retrieval, and grouping comparable email features. The strategy is based on. For all features employed in this method, a binary value of 0 or 1 is used to obtain the outcome, where 1 signifies a positive value phishing feature, and 0 if it isn't. Because this technique lacks a more dynamic system, it is less effective is less capable of producing reliable results.
Naresh, U.. (2013).	Intelligent Phishing Website Detection and Prevention System by Using Link Guard Algorithm	For hyperlinks, a system based on the link guard algorithm was proposed. The method performs tests such as comparing the DNS of actual and visual links, checking IP address dotted decimal, checking encoded links, and pattern matching. The disadvantages of this technique are that it generates false positive results if a legitimate site uses an IP address rather than a domain name, and it treats phishing sites as normal if the user does not visit the original site. As a result, incorrect negative inferences are reached.

Afroz, Sadia & Greenstadt, Rachel. (2009).	PhishZoo: Detecting Phishing Websites by Looking at Them	By comparing the content of current phishing sites to the content of authentic sites, the system discovers current phishing sites. To avoid phishing, this will match the visuals, contents, and structure of the website with a trusted one. The algorithm's disadvantages are that it requires a matching image site and is less effective at detecting phishing assaults.
Yang, Peng & Zhao, Guangzhen & Zeng, Peng. (2019)..	Phishing Website Detection Based on Multidimensional Features Driven by Deep Learning	The model conducts a series of experiments on a dataset containing millions of phishing and legitimate URLs. According to the findings, the MFPD technique is effective in terms of accuracy, false positive rate, and detection speed. Deep learning could be used to extract features from web page code and text in a future version of this approach.
Lee, Lung-Hao & Lee, Kuei-Ching & Juan, Yen-Cheng & Chen, Hsin-Hsi & Tseng, Yuen-Hsien. (2014).	Users' behavioral prediction for phishing detection	For context-aware phishing detection, this study examines consumers' web browsing behaviour when confronted with phishing circumstances. They create a linear chain CRF model for users' behavioural prediction by extracting discriminative information from each clicked URL, such as domain name, bag-of-words, generic Top-Level Domains, IP address, and port number. Experiments on a large scale reveal that this technology is capable of forecasting the phishing dangers that consumers will face in the future. The model could be enhanced by include several other variables that can be used to detect phishing.
Gowtham, Ramesh & Krishnamurthi, Ilango. (2013).	PhishTackle—a web services architecture for anti-phishing	PhishTackle is an anti-phishing web services architecture. The first tier of the AntiPhishing system is the browser extension module, which gathers URLs requested by users from the online browser and sends them to phishing verification. The second layer's middleware web services allow us to present a publishedAntiPhishing services

		front-end to client apps. The third layer, represented by An-tiPhishing components, detects phishing webpages using a collection of heuristics and resources available on the internet.
Wang, Y., Agrawal, R., & Choi, B. (2008).	Light Weight Anti-Phishing with User Whitelisting in a Web Browser	The model has proposed a light-weight user whitelist technique to provide protection for home users. The significance of this study is to identify an approach that is simple to use and effective for individual users. In addition it has shown their approach imposes little overhead on browsers and requires no modification of servers. The simplicity of the solutiondoesn't provides high security has a chance of private information leakage.

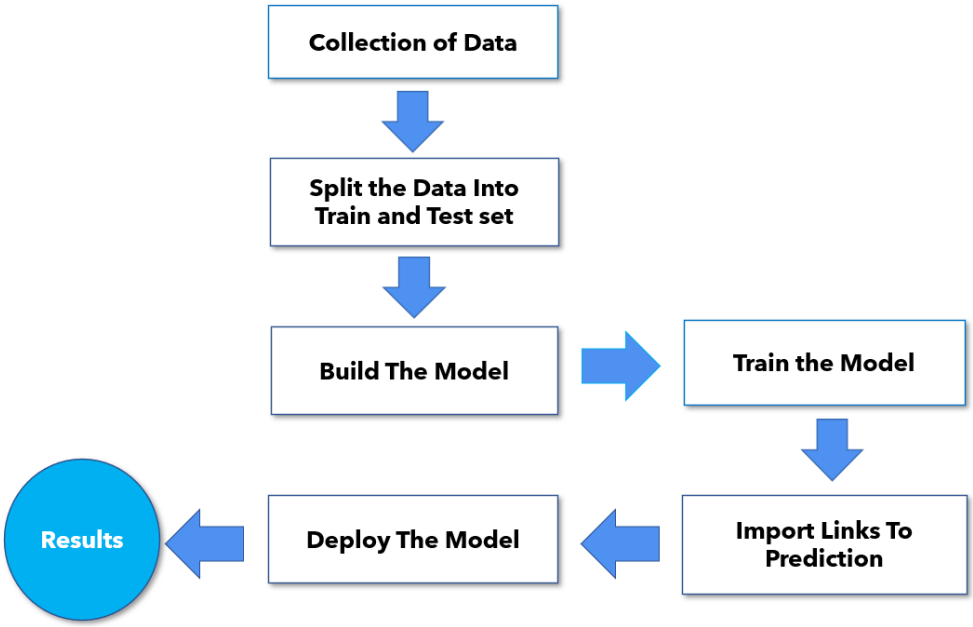
3. Overview of the work

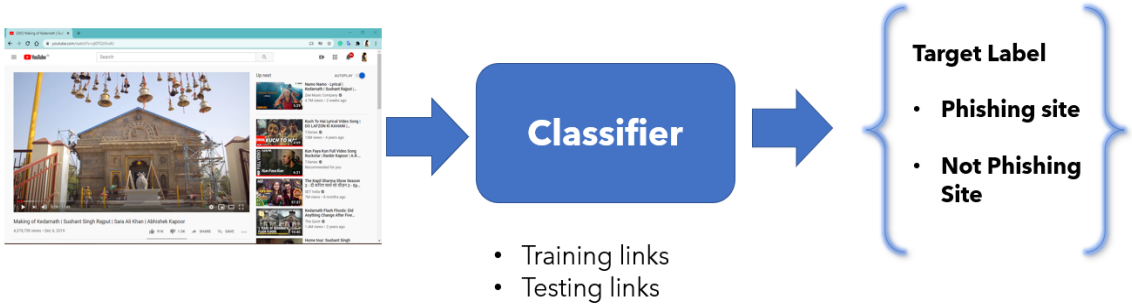
3.1. Problem description

Phishing continues to prove one of the most successful and effective ways for cybercriminals to defraud us and steal our personal and financial information. Our developing dependence on the web to lead a lot of our everyday business has furnished fraudsters with the ideal climate to dispatch designated phishing assaults. The phishing assaults occurring today are modern and progressively harder to spot. A review led by Intel saw as that 97% of safety specialists come up short at distinguishing phishing messages from certifiable messages. In any case, it's not simply malignant messages that are utilized to fool individuals into tapping on joins or revealing delicate data. One more typical strategy utilized by lawbreakers includes the making of phony sites to fool casualties into entering touchy data. Phishing sites are made to trick clueless clients into thinking they are on an authentic site. The hoodlums will invest a ton of energy causing the site to appear as valid as could be expected and many destinations will show up practically vague from the genuine article.

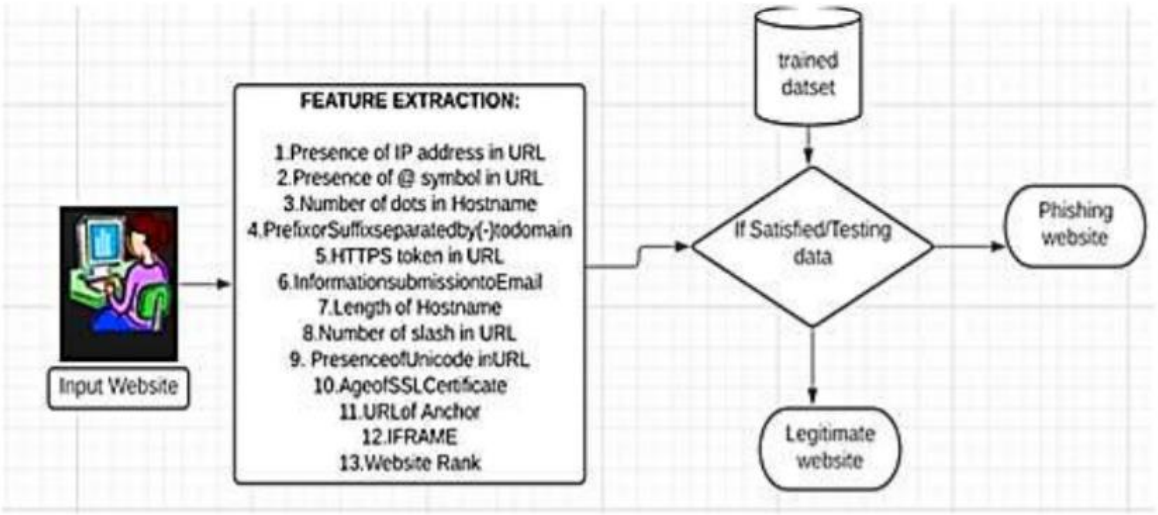
3.2. Working model

Below is the basic workflow of the entire project and methodology adopted to successfully implement the project.





3.3.Design description



Above figure shows the system architecture for detecting phishing URL. The foremost objective of this system is to identify an URL that is provided as input as a phished URL or not. The proposed method consists of two phases (1) URL search phase and (2) feature extraction phase. In the URL search phase, once the user accesses/requests an URL, a search is carried out to check whether the given URL is in the repository of legitimate URLs. If a match is found in the repository, then the URL is considered to be a legitimate URL. Otherwise, the URL is not a legitimate URL and it undergoes the next phase. The main reason for carrying out the search phase before the feature extraction phase is it reduces the unnecessary computation during the feature extraction phase and improves the overall response time of the system. In the Feature extraction phase, we have defined heuristics to extract 14 features from the URL and are subjected to association rule mining to determine the legitimate and phished URL.

3.4. Algorithms

- Logistic Regression

  Logistic Regression is a Machine Learning classification algorithm that is used to predict the probability of a categorical dependent variable. In logistic regression, the dependent variable is a binary variable that contains data coded as 1 (yes, success, etc.) or 0 (no, failure, etc.). In other words, the logistic regression model predicts P(Y=1) as a function of X.

  It comes under the Supervised Learning technique and is used for predicting the categorical dependent variable using a given set of independent variables.

  Logistic regression predicts the output of a categorical dependent variable. Therefore, the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1. Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas Logistic regression is used for solving the classification problems. In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1). The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, a mouse is obese or not based on its weight, etc. Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete datasets. Logistic Regression can be used to classify the observations using different types of data and can easily determine the most effective variables used for the classification.

- MultinomialNB

  Applying Multinomial Naive Bayes to NLP Problems. Naive Bayes Classifier Algorithm is a family of probabilistic algorithms based on applying Bayes' theorem with the “naive” assumption of conditional independence between every pair of a feature.

  It is a grouping strategy dependent on Bayes' Theorem with a supposition of

freedom among indicators. In straightforward terms, a Naive Bayes classifier accepts that the presence of a specific element in a class is disconnected to the presence of some other component. For instance, an organic product might be viewed as an apple on the off chance that it is red, round, and around 3 inches in width. Regardless of whether these highlights rely upon one another or upon the presence of different highlights, these properties freely add to the likelihood that this natural product is an apple and that is the reason it is known as 'Naive'. Naive Bayes model is anything but difficult to construct and especially helpful for exceptionally enormous informational collections. Alongside straightforwardness, Naive Bayes is known to beat even exceptionally refined arrangement techniques.

- Random Forest Classifier

  Random forest algorithm is one of the most powerful algorithms in machine learning technology and it is based on concept of decision tree algorithm. Random forest algorithm creates the forest with number of decision trees. High number of trees gives high detection accuracy. Creation of trees are based on bootstrap method. In bootstrap method features and samples of dataset are randomly selected with replacement to construct single tree. Among randomly selected features, random forest algorithm will choose best splitter for the classification and like decision tree algorithm; Random Forest algorithm also uses Gini index and information gain methods to find the best splitter. This process will get continue until random forest creates n number of trees. Each tree in forest predicts the target value and then algorithm will calculate the votes for each predicted target. Finally, random forest algorithm considers high voted predicted target as a final prediction.

4. Advantages and Disadvantages of Previously Used Methods

Blacklists: Blacklists hold URLs (or parts thereof) that refer to sites that are considered malicious. Whenever a browser loads a page, it queries the blacklist to determine whether the currently visited URL is on this list. If so, appropriate countermeasures can be taken. Otherwise, the page is considered legitimate. The blacklist can be stored locally at the client or hosted at a central server.

Advantages: Obviously, an important factor for the effectiveness of a blacklist is its coverage. The coverage indicates how many phishing pages on the Internet are included in the list. Another factor is the quality of the list. The quality indicates how many non-phishing sites are incorrectly included into the list. For each incorrect entry, the user experiences a false warning when she visits a legitimate site, undermining her trust in the usefulness and correctness of the solution. Finally, the last factor that determines the effectiveness of a blacklist-based solution is the time it takes until a phishing site is included. This is because many phishing pages are short-lived and most of the damage is done in the time span between going online and vanishing. Even when a blacklist contains many entries, it is not effective when it takes too long until new information is included or reaches the clients.

Disadvantages: The overall technique to identify phishing sites by refreshing boycotted URLs, Internet Protocol (IP) to the antivirus information base which is otherwise called "boycott" strategy. To dodge boycotts, assailants utilizes innovative procedures to trick clients by adjusting the URL to seem real through muddling and numerous other basic methods including: quick motion, in which intermediaries are naturally produced to have the page; algorithmic age of new URLs; and so forth. Significant disadvantage of this strategy is that it can't recognize Zero-hour phishing attack.

5. Implementation

5.1. Description of Modules/Programs

- Importing Libraries

```
import pandas as pd # use for data manipulation and analysis
import numpy as np # use for multi-dimensional array and matrix

import seaborn as sns # use for high-level interface for drawing attractive and informative statistical graphics
import matplotlib.pyplot as plt # It provides an object-oriented API for embedding plots into applications
%matplotlib inline
# It sets the backend of matplotlib to the 'inline' backend:
import time # calculate time

from sklearn.linear_model import LogisticRegression # algo use to predict good or bad
from sklearn.naive_bayes import MultinomialNB # nlp algo use to predict good or bad
from sklearn.ensemble import RandomForestClassifier

from sklearn.model_selection import train_test_split # spliting the data between feature and target
from sklearn.metrics import classification_report # gives whole report about metrics (e.g, recall,precision,f1_score,c_m)
from sklearn.metrics import confusion_matrix # gives info about actual and predict
from nltk.tokenize import RegexpTokenizer # regexp tokenizers use to split words from text
from nltk.stem.snowball import SnowballStemmer # stemmes words
from sklearn.feature_extraction.text import CountVectorizer # create sparse matrix of words using regexptokenizes
from sklearn.pipeline import make_pipeline # use for combining all prerocessors techniuqes and algos

from PIL import Image # getting images in notebook
# from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator# creates words colud
```

```
from bs4 import BeautifulSoup # use for scraping the data from website
# from selenium import webdriver # use for automation chrome
import networkx as nx # for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks.

import pickle# use to dump model

import warnings # ignores pink warnings
warnings.filterwarnings('ignore')
```

- Loading Dataset

```
phish_data = pd.read_csv('phishing_site_urls.csv')
```

- **About dataset**
- Data is containg 5,49,346 unique entries.
- There are two columns.
- Label column is prediction col which has 2 categories A. Good - which means the urls is not containing malicious stuff and **this site is not a Phishing Site.** B. Bad - which means the urls contains malicious stuffs and **this site isa Phishing Site.**
- There is no missing value in the dataset.

- Data Preprocessing

  After collecting the data, we have to vectorize our URLs. We used Count Vectorizer and gather words using tokenizer, since there are words in URLs that are more important than other words e.g., ‘virus’, ‘.exe’,’.dat’ etc.

  Converting the URLs into a vector form using Regexp tokenizer.

```
tokenizer = RegexpTokenizer(r'[A-Za-z]+')


phish_data.URL[0]
```

Also using tools like snowball stemmer and Beautiful Soap for preprocessing.

- Visualization

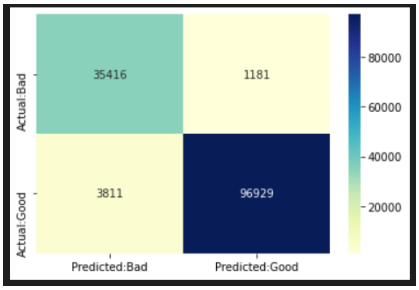
  Using Word Cloud for visualizing important hidden words and keys

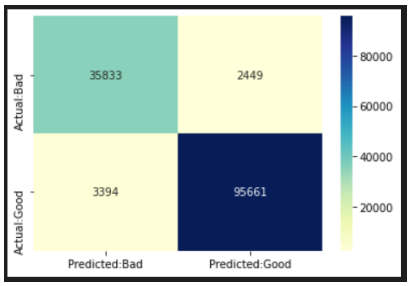
```
#sliceing classes
bad_sites = phish_data[phish_data.Label == 'bad']
good_sites = phish_data[phish_data.Label == 'good']
```

- Creating ML model using ML algorithms

  1. Random Forest
  2. Multinomial NB
  3. Logistics Regression

- Forming Confusion Matrix and determining which algorithm works best





- Integration and deployment through FAST API which shows result if the website URL is phishing or not.

```python
import uvicorn
from fastapi import FastAPI
import pickle

app = FastAPI()

#pkl
phish_model_ls = pickle.load(open('phishing.pkl', 'rb'))

# ML Aspect
@app.get('/predict/{feature}')
async def predict(features):
    X_predict = []
    X_predict.append(str(features))
    y_Predict = phish_model_ls.predict(X_predict)
    if y_Predict == 'bad':
        result = "This is a Phishing Site"
    else:
        result = "This is not a Phishing Site"

    return (features, result)
if __name__ == '__main__':
    uvicorn.run(app,host="127.0.0.1",port=8000)
```

5.2. Algorithms source code

5.2.1. Logistics Regression

```python
Scores_ml = {}
Scores_ml['Logistic Regression'] = np.round(lr.score(testX,testY),2)


print('Training Accuracy :',lr.score(trainX,trainY))
print('Testing Accuracy :',lr.score(testX,testY))
con_mat = pd.DataFrame(confusion_matrix(lr.predict(testX), testY),
            columns = ['Predicted:Bad', 'Predicted:Good'],
            index = ['Actual:Bad', 'Actual:Good'])


print('\nCLASSIFICATION REPORT\n')
print(classification_report(lr.predict(testX), testY,
                            target_names =['Bad','Good']))

print('\nCONFUSION MATRIX')
plt.figure(figsize= (6,4))
sns.heatmap(con_mat, annot = True,fmt='d',cmap="YlGnBu")
```

```
Training Accuracy : 0.9782480479795345
Testing Accuracy : 0.9636514559077306
```

5.2.2. Multinomial NB

```
Scores_ml['MultinomialNB'] = np.round(mnb.score(testX,testY),2)
```

```
print('Training Accuracy :',mnb.score(trainX,trainY))
print('Testing Accuracy :',mnb.score(testX,testY))
con_mat = pd.DataFrame(confusion_matrix(mnb.predict(testX), testY),
                       columns = ['Predicted:Bad', 'Predicted:Good'],
                       index = ['Actual:Bad', 'Actual:Good'])


print('\nCLASSIFICATION REPORT\n')
print(classification_report(mnb.predict(testX), testY,
                            target_names =['Bad','Good']))

print('\nCONFUSION MATRIX')
plt.figure(figsize= (6,4))
sns.heatmap(con_mat, annot = True,fmt='d',cmap="YlGnBu")
```

```
Training Accuracy : 0.9741437687040817
Testing Accuracy : 0.9574550194048217
```

5.2.3. Random Forest

```
Scores_ml = {}
Scores_ml['Random Forest'] = np.round(lr.score(testX,testY),2)
```

```
print('Training Accuracy :',lr.score(trainX,trainY))
print('Testing Accuracy :',lr.score(testX,testY))
con_mat = pd.DataFrame(confusion_matrix(lr.predict(testX), testY),
                       columns = ['Predicted:Bad', 'Predicted:Good'],
                       index = ['Actual:Bad', 'Actual:Good'])


print('\nCLASSIFICATION REPORT\n')
print(classification_report(lr.predict(testX), testY,
                            target_names =['Bad','Good']))

print('\nCONFUSION MATRIX')
plt.figure(figsize= (6,4))
```

```
Training Accuracy : 0.9741437687040817
Testing Accuracy : 0 0.98 50194048217
```

5.3. Tools

- Snowball Stemmer

  Snowball is a small string processing language, gives root words

```
stemmer = SnowballStemmer("english") # choose a language
```

- Regexp Tokenizer

  A tokenizer that splits a string using a regular expression, which matches either the tokens or the separators between tokens.

```
tokenizer = RegexpTokenizer(r'[A-Za-z]+')
```

- Chrome web driver

  WebDriver tool use for automated testing of webapps across many browsers. It provides capabilities for navigating to web pages, user input and more.

```
browser = webdriver.Chrome(r"chromedriver.exe")
```

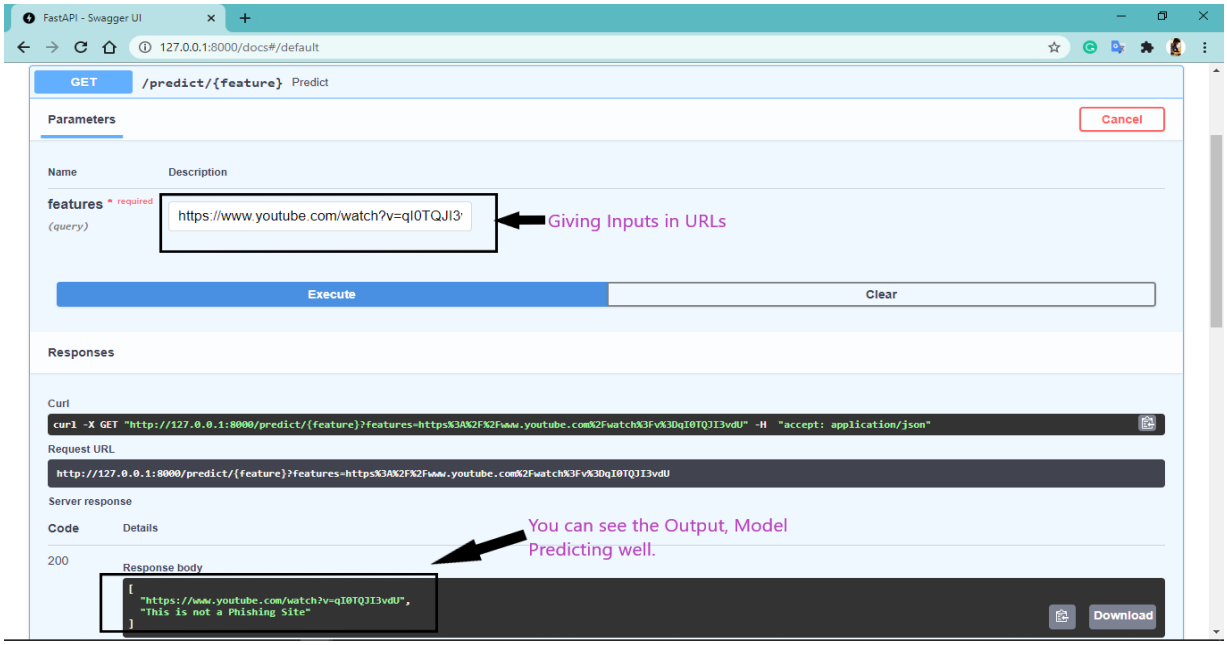
- Beautiful Soup

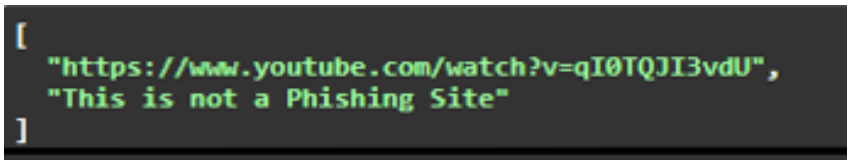
  It is use for getting data out of HTML, XML, and other markup languages.

```
for url in list_urls:
    browser.get(url)
    soup = BeautifulSoup(browser.page_source,"html.parser")
    for line in soup.find_all('a'):
        href = line.get('href')
        links_with_text.append([url, href])
```

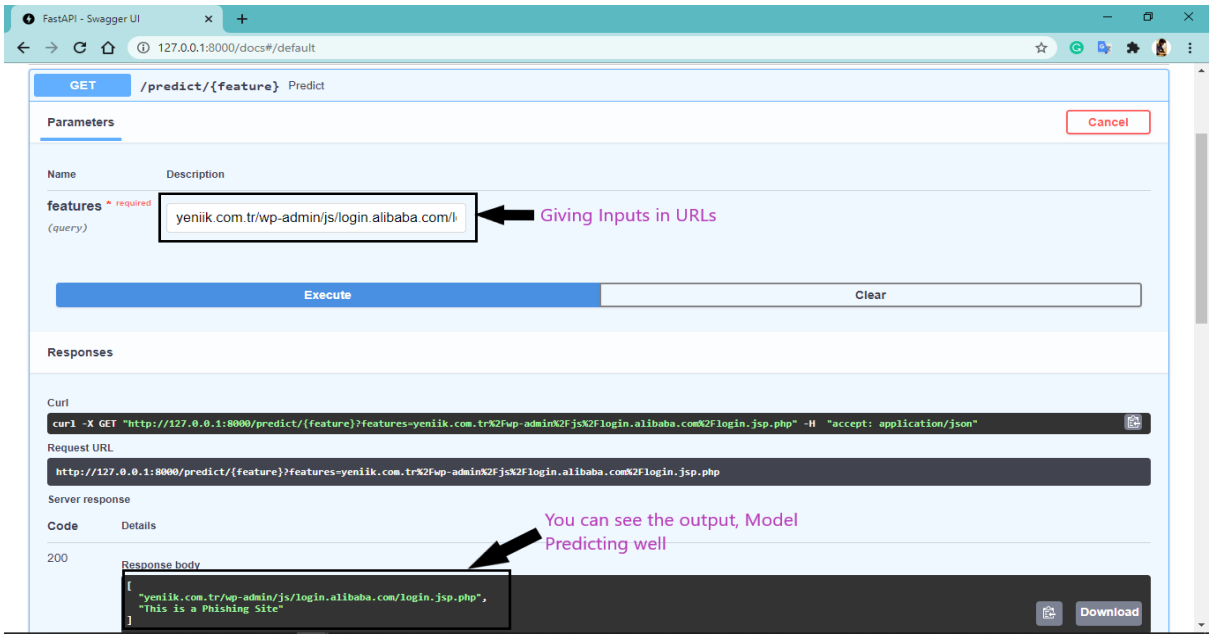
5.4. Execution of the project

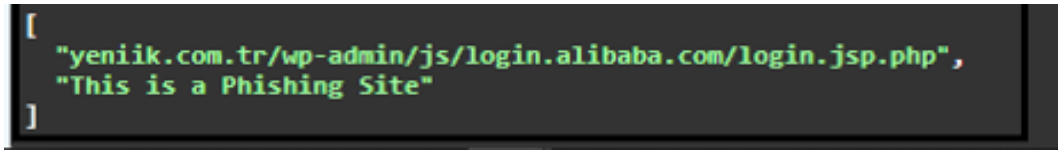
Case 1. Input a URL which is not a phishing website.





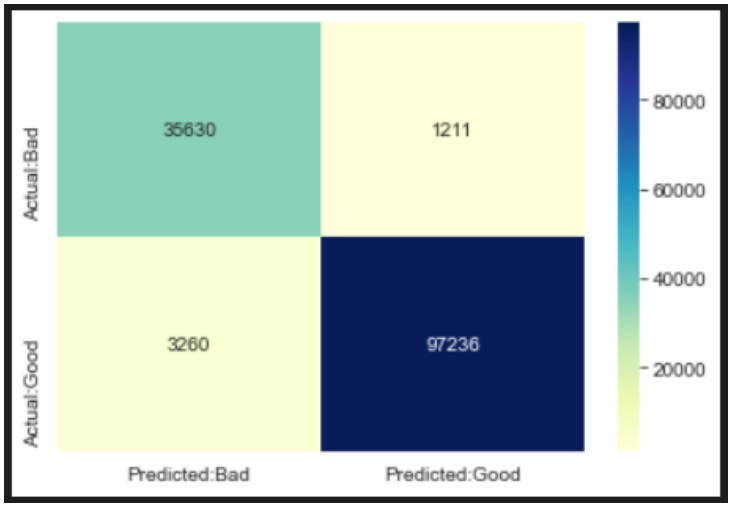
Case 2. Input a URL which is a phishing website.

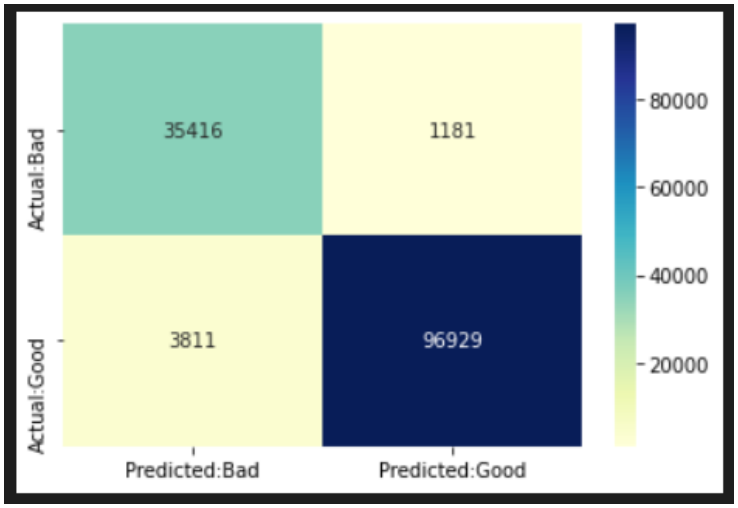


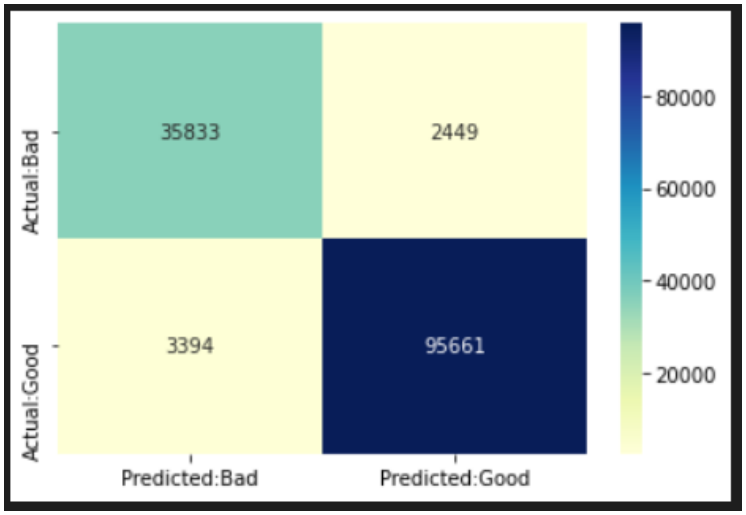


5.5.Results analysis

Confusion Matrix of the three algorithms







6. Conclusion and Future Scope

This project intends to upgrade recognition technique to recognize phishing sites utilizing machine learning innovation. From our experiment we found Random Forest Classifier has the highest accuracy rate as 98% as compared to MultinomialNB (95%) and Logistic Regression (96%). Likewise result shows that classifiers give better execution when we utilized more information as preparing information. The integration with FAST API also grants the user to see precise and quick result whether the website URL is a phishing or not .In future half breed innovation will be executed to distinguish phishing sites more precisely, for which random forest algorithm of machine learning innovation and boycott technique will be utilized. Even though Logistic Regression is as close to random forest we will prefer random forest for detecting the phishing sites. Hence the users can get rid of phishing sites and be safe by avoiding them using this technique.

7. REFERENCES

[1] Chandrasekaran, Madhusudhanan, Krishnan Narayanan, and Shambhu Upadhyaya. "Phishing email detection based on structural properties." NYS Cyber Security Conference. 2006.

[2] Wenyin, Liu, et al. "Discovering phishing target based on semantic link network." Future Generation Computer Systems 26.3 (2010): 381- 388.

[3] Almomani, Ammar, et al. "Evolving fuzzy neural network for phishing emails detection." Journal of Computer Science 8.7 (2012): 1099.

[4] Madhuri, M., K. Yeseswini, and U. Vidya Sagar. "Intelligent phishing website detection and prevention system by using link guard algorithm." Int. J. Commun. Netw. Secur 2 (2013): 9-15.

[5] Afroz, Sadia, and Rachel Greenstadt. "Phishzoo: Detecting phishing websites by looking at them." Semantic Computing (ICSC), 2011 Fifth IEEE International Conference on. IEEE, 2011.

[6] Yang, Peng & Zhao, Guangzhen & Zeng, Peng. (2019). Phishing Website Detection Based on Multidimensional Features Driven by Deep Learning. IEEE Access. PP. 1-1. 10.1109/ACCESS.2019.2892066.

[7] Lee, Lung-Hao & Lee, Kuei-Ching & Juan, Yen-Cheng & Chen, Hsin-Hsi & Tseng, Yuen-Hsien. (2014). Users' behavioral prediction for phishing detection. 337-338. 10.1145/2567948.2577320.

[8] Gowtham, Ramesh & Krishnamurthi, Ilango. (2013). PhishTackle—a web services architecture for anti-phishing. Cluster Computing. 17. 10.1007/s10586-013-0320-5.

[9] Wang, Yue & Agrawal, Rinky & Choi, Baek-Young. (2008). Light Weight Anti-Phishing with User Whitelisting in a Web Browser. 1 - 4. 10.1109/TPSD.2008.4562720.

[10] Mao, Jian & Li, Pei & Wei, Tao & Liang, Zhenkai. (2017). Phishing-Alarm: Robust and Efficient Phishing Detection via Page Component Similarity. IEEE Access. PP. 1-1. 10.1109/ACCESS.2017.2743528.