REPORT

PROJECT WORK FOR COURSE

“MACHINE LEARNING FOUNDATION”

(INT 247)

(Malicious URL Detection using Machine Learning)

Name : Rohit Kumar Sharma

Roll No : RKM012A01

Registration Number : 12012980

Degree: B.Tech CSE

Semester: 6th

School: School of Computer Science and Engineering

University: Lovely Professional University

GitHub Link: https://github.com/rohit133/12012980-Malicious-URL-Detection-using-Machine-Learning

Malicious URL Detection using Machine Learning

INTRODUCTION:

In today's world, there is a rapid advancement in technology. With the advancement of technology, there's a similar development in the Internet. Internet involvement in social and business fields is increasing in large scale. The increasing use of the internet for such purposes increases the scope for cyber-criminal activities. As the connectivity and the number of users grow, there is a proportional increase in attackers. The Government, industry and individuals are the victims. It is a difficult task to predict the future threats and their nature, and practically unsolvable. Malware or malicious websites become one of the major threat for cyber security. Whereas malicious URLs becomes a serious threat of cyber security. Malicious URL is a common and serious threat to cyber-security. Malicious URLs host content abnormalities, such as spamming, phishing attacks, exploiting users, etc. They allow unsuspected users as victims of attacks by drivers. They incur huge monetary loss of billions of dollars every year worldwide. It is very important to firstly detect and act on such attacks frequently for security Generally, such detections are done using big blacklists In practice, it is not possible to have exhaustive blacklists. Today's naive implementation of detection techniques is insufficient to address billions of URLs encountered in everyday life. Machine Learning techniques is used to address the problem as a binary classification problem in large scale. There are various classifiers in Machine Learning which give high accuracy in classification of good and bad URLs Moreover, Huang et al. detects Malicious URL using a greedy selection algorithm Similarly, Liu et al. also provides experimental study on URL detection using Machine Learning algorithms. Vu et al. performs cost-sensitive malicious URL detection using a Decision Tree algorithm.

OBJECTIVE:

In this project admis to classify the URL using various Machine learning model such as Logistic regression, Decision Tree, Random forest Gaussian Naive Bayes, Ada Boosting and Gradient Boosting using these model on dataset. Datasets contain two fields "*URL*" and "*Class"* and after that extracting the features from the URL class features extraction is divided into 3 parts such as lexical, Host and other using the "*URL*". After obtaining the features the features are passed into the machine learning model and obtain the accuracy after comparing the accuracy of the models the highest accuracy wins. Then passing the testing data to the winner model and predicting the weather the URL is malicious or not.

MACHINE LEARNING ALGORTHIMS USED:

Decision Tree
Random Forest
Adaboost
Gradient Boosting
GNB
Logistic Regression

PROPOSED MODEL DIAGRAM:

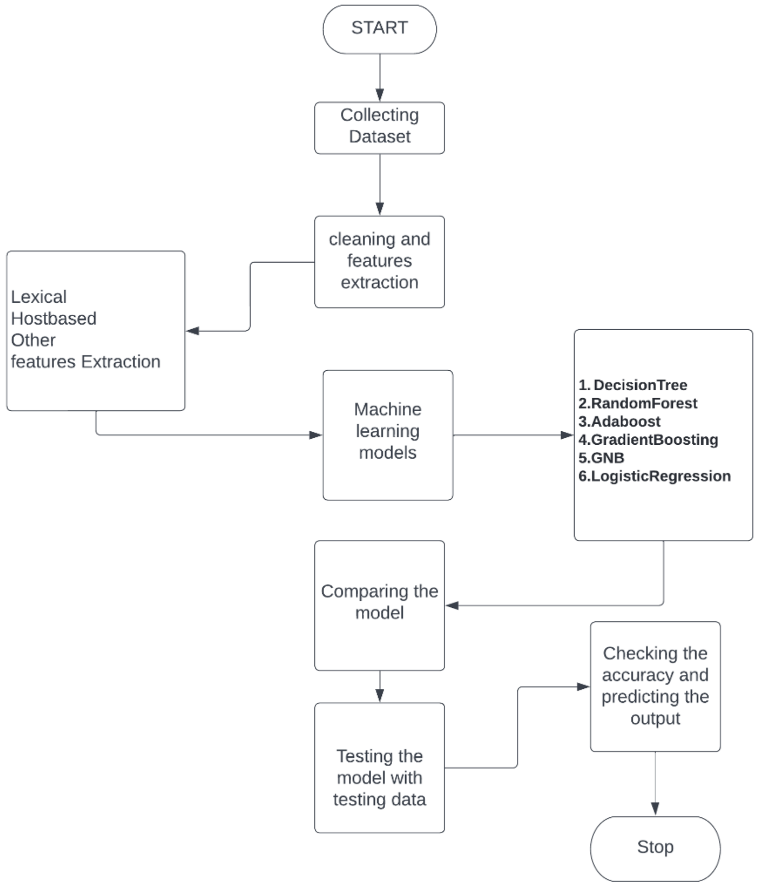

Figure 1

IMPLEMENTATION:

Data is loaded using Pandas module. This dataset contains around 1056937 rows of data in which around 56937 rows are classified as bad URLs, so I created a subset of dataframe that contain equal numbers of ‘good’ and ‘bad’ URLs.

Training Url set

```python
df = pd.read_csv("https://raw.githubusercontent.com/rlilojr/Detecting-Malicious-URL-Machine-Learning/master/dataset.csv")
df.groupby('label').count()
```

[81] ✓ 7.8s Python

	url
label	
0	1000000
1	56937

Tranining Urls and Model Feeding Set

```python
data_test=df.head(113874)
data_test.groupby('label').count()
```

[83] ✓ 0.1s Python

	url
label	
0	56937
1	56937

Then the dataframe passed through multiple features extracting function that gain information using 3rd party Libraries such as ‘tldextract’, ‘urlparse’ and ‘splitext’ to gain information about the host.

```python
def dotCount(url):
    return url.count('.')
```

[85] ✓ 0.4s Python

```python
def delimCount(url):
    delim=[';','_','?','=','&']
    for item in url:
        if item in delim:
            count +=1
    return count
```

[86] ✓ 0.7s Python

```python
def isip(uri):
    try:
        if ip.ip_address(uri):
            return 1
    except:
        return 0
```

[87] ✓ 0.5s Python

```python
def ifHyphen(url):
    return url.count('-')
```

[88] ✓ 0.7s Python

```python
def ifAt(url):
    return url.count('@')
```

[89] ✓ 0.7s Python

Then using a separate function to call all the other features extracting function and storing their values in to a dataframe.

```python
def getFeatures(url, label):
    result = []
    url = str(url)

    # Adding inputs to the 'result' list
    result.append(url)

    path = urlparse(url)
    ext = tldextract.extract(url)

    result.append(dotCount(ext.subdomain))

    result.append(ifHyphen(path.netloc))

    result.append(len(url))

    result.append(ifAt(path.netloc))

    result.append(ifDoubleSlash(path.path))

    result.append(countSubDir(path.path))

    result.append(countSubDomain(ext.subdomain))

    result.append(len(path.netloc))

    result.append(len(path.query))

    result.append(isip(ext.domain))

    result.append(1 if ext.suffix in Suspicious_TLD else 0)

    result.append(1 if '.'.join(ext[1:]) in Suspicious_Domain else 0)

    result.append(str(label))
    return result
```

0.5s

Python

Now passing the data into the function to extract information about the website.

```python
for i in range(len(data_test)):
    features = getFeatures(data_test['url'].loc[i], data_test['label'].loc[i])
    featureSet.loc[i] = features
```

15m 2.1s

Python

```python
featureSet
```

0.1s

Python

	url	no of dots	presence of hyphen	len of url	presence of at	presence of double slash	no of subdir	no of subdomain	len of domain	no of queries	is IP	presence of Suspicious_TLD	presence of suspicious domain	label
0	http://br-ofertasimperdiveis.epizy.com/produto...	0	1	186	0	0	1	1	31	135	0	0	0	1
1	https://semana-da-oferta.com/produtos.php?id=5...	0	2	58	0	0	1	0	20	16	0	0	0	1
2	https://scrid-apps-creacust-sslhide90766752024...	0	4	72	0	0	2	1	53	0	0	0	0	1
3	http://my-softbank-security.com/wap_login.htm	0	2	45	0	0	1	0	24	0	0	0	0	1
4	http://www.my-softbank-security.com/wap_login.htm	0	2	49	0	0	1	1	28	0	0	0	0	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
113869	u-cergy.fr	0	0	10	0	0	0	0	0	0	0	0	0	0
113870	mahindra.com	0	0	12	0	0	0	0	0	0	0	0	0	0
113871	weddingplanner-net.com	0	0	22	0	0	0	0	0	0	0	0	0	0
113872	speedkartsp.com.br	0	0	18	0	0	0	0	0	0	0	0	0	0
113873	biznes-trainer.ru	0	0	17	0	0	0	0	0	0	0	0	0	0

113874 rows × 14 columns

Once the features are extracted I applied some Visualization in order to obtain some insight on the data.

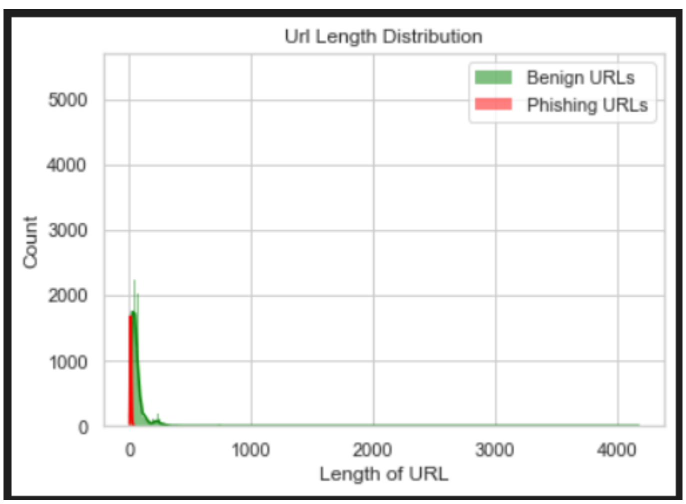

Figure 2: Checking the length of the Benign URLs and Phishing URLs

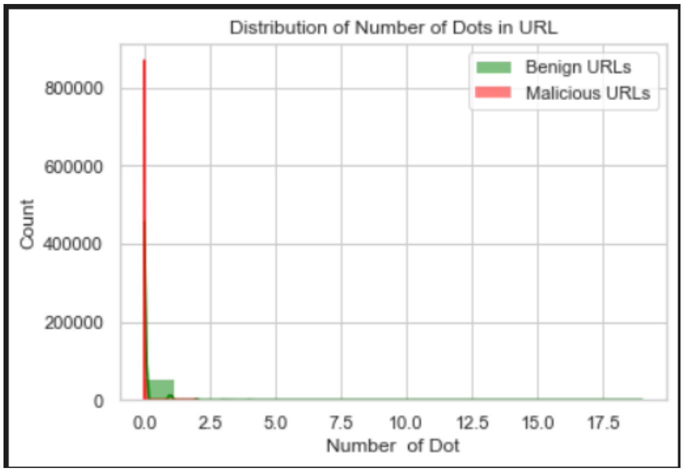

Figure 3: Comparing the number of dots in between Benign and Malicious URLs

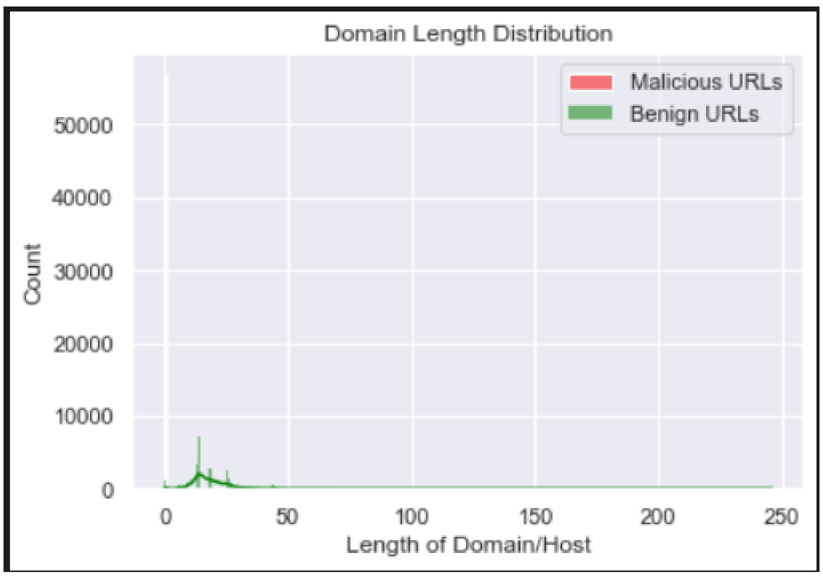

Figure 3: Check the length of Domain

Now applying the machine learning model dividing my working data into training and testing data using train_test_split with test_size =0.3 and random_stauts = 44.

Then applying the various machine learning models into onto the training data and comparing their accuracy score and storing it into the winner variable then applying the winner model on testing data and checking its confusion matrix .

Model Name	**Accuracy Score**
DecisionTree	0.9935895559523461
RandomForest	0.9935895559523461
Adaboost	0.9933261130462782
GradientBoosting	0.9936480988203612

GNB	0.9911892983637268
LogisticRegression	0.9936773702543688

Table 1 Comparison of Models

LogisticRegression :	**0.9936773702543688**

Table 2 Winner Model

	precision	recall	f1-score	support
0	**0.99**	**1.00**	**0.99**	**56937**
1	**1.00**	**0.99**	**0.99**	**56937**

Table 3 Precision and Recall

Accuracy			**0.99**	**113874**
Macro Avg	**0.99**	**0.99**	**0.99**	**113874**
Weighted Avg	**0.99**	**0.99**	**0.99**	**113874**

Table 4 Accuracy Score

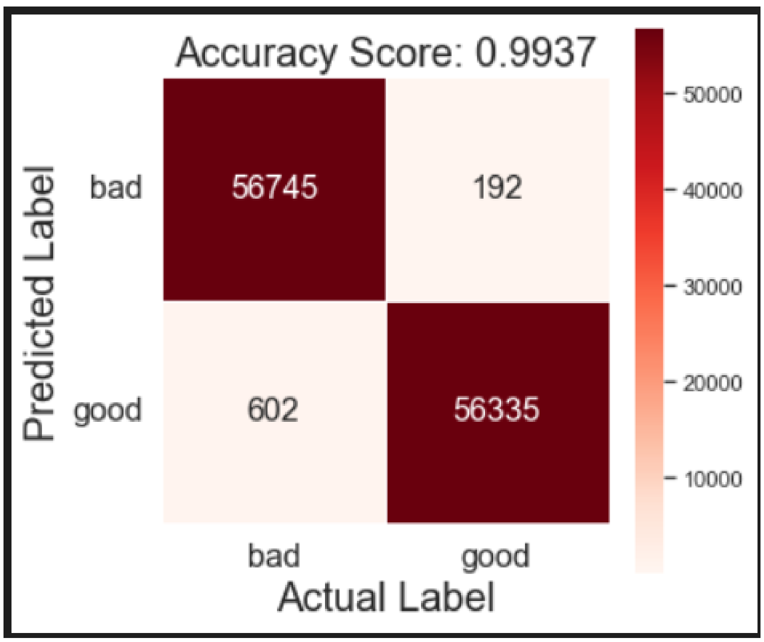

Figure 4 Confusion Matrix for the Logistic Regression

CONCLUSION:

Malicious Web sites are the basis of most of the criminal activities over the internet. The dangers that arise due to the malicious sites are enormous and the end-users must be prohibited from visiting such sites. The users should prohibit themselves from clicking on such Uniform Resource Locator (URL). The detection of malicious URLs is a binary classification problem and several Machine Learning Algorithms, namely Random Forests, SVMs and Naïve Bayes are implemented on training dataset. Also, it has been seen that the logistic Regression with 0.993677% of accuracy. performs better from any other classification models.