Data and Web Mining Report

Malicious URLS Detection

Submitted by(Batch-1):

Avvari Likhitha(AP19110010171)
Gaddam Venkata Siva Sai Nishita(AP19110010302)
Attuluri Prudhvi Raj(AP19110010234)
Danda Sai Praveen(AP19110010215)
Muvva George Stephen(AP19110010151)

Department of Computer Science and Engineering
SRM University-AP, Andhra Pradesh, India

May- 2022

Abstract:

One of the most frequent cybersecurity concerns is malicious unified resource locator (URL), also known as Malicious websites. They host malicious content (spam, malware, inappropriate ads, spoofing, and so on) and lure unsuspecting users into becoming fraud victims (loss of money, personal information disclosure, ransomware installation, extortion, falsified shopping website, surprising prize, and so on), likely to result in Millions of rupees in losses each year. Emails, promotions, online searches, and referrals from other sites can all increase traffic to these websites. The user has to click on the malicious link in each instance. The increase in phishing, spamming, and malware attacks have led to the evolution of a reliable solution that can analyze and identify harmful URLs.

Introduction:

The structure and format of a URL are particular. To manipulate visitors and distribute their malicious URL, attackers frequently try to modify one or more components of the URL's structure. Malicious URLs are links that have a negative impact on users. These URLs will send visitors to resources or pages where attackers can have full control over the site, and redirect them to desirable sites. They'll get to know our login credentials or they can have complete access over our device. Malicious URLs can also be concealed in seemingly secure download links, and can swiftly propagate through file and message exchange across shared networks. Drive-by Download, Phishing and Social Engineering, and Spam are some of the attack tactics that exploit malicious URLs.

Technologies used:

1. Python
2. Machine Learning
3. Data Mining
4. Data Analysis

Problem Survey:

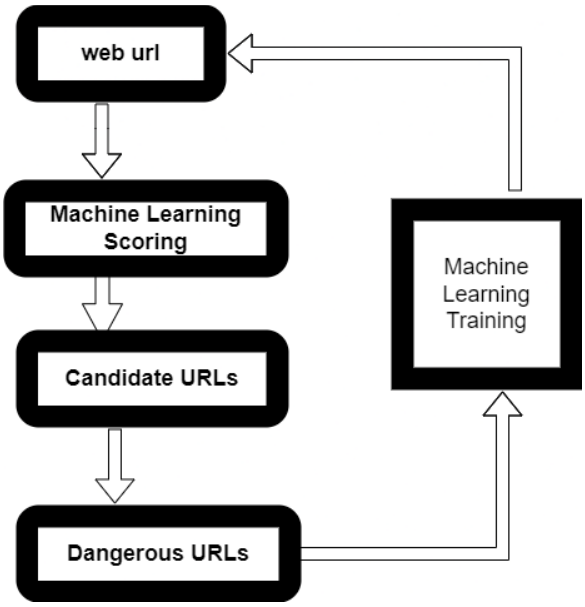
What are the problems & consequences encountered by opening or visiting a malicious URL without knowing about it? From an online survey, we found out that almost 4.95 billion people in the world use the internet which is equal to 62.8 percent of the world's entire population. This data of users has been never decreasing, it has been increasing from day to day. As the coronavirus pandemic has had a big effect on internet usage the actual growth figures may be much higher than this. So there are many users who spend every day online for their business, learning new things, browsing social media websites, exploring for new information. It's become so commonplace that we usually ignore just how harmful the internet can be for the users. While

there are numerous legitimate websites out there with effective cybersecurity measures in place, there are also many malicious websites URLs that are created exclusively to cause harm. In easy words, a malicious URL is a malicious link that is made with the purpose of enabling attacks, frauds, and scams. By clicking on an infected URL, it can download malware such as a Trojan attack or virus that can take possession of your devices, or one can be induced to provide sensitive details on a fake website. As the title implies, nothing useful can ever reach out of a malicious URL. The possible harm is so immense that malicious links are regarded as one of the biggest threats to the digital world, particularly when we speak regarding attacks and dangers in the internet world we are living in. As an outcome of this issue, many studies & approaches have come into the picture for this detection of malicious URLs.The development of this technology of malicious URL detection can help users identify malicious URLs and stop users from being attacked by malicious URLs.So with this approach many users can be helped in identifying the URLs are malicious or not, also the malicious URLs detection helps numerous individuals from not falling a victim to cybercrime.

Literature review:

Existing Solution:

Described a summary of the current condition of malware on the internet. The assessment is based on Internet-wide data taken during a 12-month period beginning in March 2006. Several attack tactics for converting websites into malware infection vectors are revealed in the findings. Advertising, third-party widgets, user-contributed material, and web server security were identified as four main components of content management that enable browser exploitation. The paper demonstrates how each of these categories may be used to attack web browsers through analysis and demonstrations.



Proposed Model:

Two primary methods are used to identify dangerous URLs. The first is the Multinomial Naive Bayesian algorithm, which is a link analysis method, and the second is Logistic Regression algorithm. The URLs are used as labels, which are then taught

to the system. This labeling aids in determining the URL's authenticity. When a user feeds URLs into the system, they are routed through feature extraction and then to the prediction phase, where they are identified as harmful or benign URLs. When URLs are entered, they are compared against labels in the database of labeled URLs. If they are found, they are passed to the model training step, where they are classified as malicious or non-malicious. The results are obtained once they've been detected. Applying two algorithms to the dataset, comparing each algorithm by using performance measures like root mean square error.

Dataset description:

The dataset represents the classification of urls into malicious or good urls. It impacts in our daily life like stealing our personal information like installing malware and hacking into our systems so it's good to have a best classifier to classify whether the given link is malicious or not.

Representing the dataset in two columns and 549346 unique entries. The two features are url and Label, url contains different types of urls which are malicious as well as good. The label is a prediction class which contains two values.

- Good: This represents the urls not containing any malicious software behind the link or any third party websites hidden.
- Bad: This contains other third party applications or a connection between hacker and you to steal the data.

Dataset which we are taking does not contain any null values or missing values.

Data Pre-Processing:

There are 4 steps to pre-process the data accordingly: we preprocess our data according to the given problem and fit our preprocessed data to the models.

- **Data cleaning:** Dataset does not contain any missing values or any null values so the data is perfectly good for proceeding to other steps.
- **Data Integration:** At present, only one dataset is taken for testing, if required, we are going to add a few more datasets.
- **Data Transformation:** In this step we have to process the dataset into tokens and perform certain transformations to get out the root words so we undergo transformation in two steps.
    - We use **tokenizer** to split the url whenever it encounters the stopping characters like ’/’ , ‘.’ etc. Next we are going

to extract some root words by using **SnowballStemmer** in the urls that are important to compare.
- We **vectorize** the tokenized dataset into numbers.

Proposed model:

Logistic Regression:

The Logistic Regression technique is frequently employed in binary classification situations, where occurrences typically result in one of two values: pass or fail, true or false.It's best used in cases when you need to anticipate the likelihood that the dependent variable will fall into one of two response groups.
This algorithm's common use cases include determining if given handwriting matches the individual in question and whether oil prices will rise within the following months.Logistic regression is used to estimate the likelihood of a categorical dependent variable. The dependent variable in logistic regression is a binary variable that comprises data coded as 1 (yes, success, etc.) or 0 (no) (no, failure, etc.). In other words, as a function of X, the logistic regression model predicts P(Y=1).
However, unlike conventional linear regression, logistic regression's goal value is a binary variable rather than a continuous value in its most basic form.Logistic Regressions may be utilized in a variety of real-world applications, including.

- Estimating the likelihood of a person suffering a heart attack.
- Predicting if a consumer will buy a product or cancel a subscription.
- Predicting the likelihood of a process or product failing.
- Credit Scoring,
- Cancer Detection.

Working:

The linear regression model and the logistic regression equation are very equivalent.
Assume a model with one predictor "x" and one Bernoulli response variable “ŷ” and p represents the probability of f ŷ=1. The linear equation is expressed as follows:
$p = b_0+b_1x$ --------> eq 1
The RHS of the argument (b0+b1x) is a linear equation that can store values that are beyond the range of the equation (0,1).We do know, however, that frequency will always be in the range of (0,1).
To get around this, we anticipate odds rather than probability.
Odds are the ratio of the likelihood of an incident to the likelihood of it not occurring.
Odds = p/(1-p)
The first equation may be rewritten as follows:
$p/(1-p) = b_0+b_1x$ --------> eq 2
Because odds can only have a positive value, we forecast the

logarithm of odds to deal with negative quantities.
Log of odds = ln(p/(1-p))
Equation 2 may be rewritten as follows:
$\ln(p/(1-p)) = b_0+b_1x$ --------> eq 3

We use exponential on both sides to extract p from equation 3.
exp(ln(p/(1-p))) = exp(b0+b1x)
eln(p/(1-p)) = e(b0+b1x)

The inverse rule of logarithms tells us that
p/(1-p) = e(b0+b1x)

Simple algebraic manipulations
p = (1-p) * e(b0+b1x)
p = e(b0+b1x)- p * e(b0+b1x)

On the right-hand side, we'll use the letter p**.**
$p = p * ((e^{(b0+b1x)})/p - e^{(b0+b1x)})$
$p = e^{(b0+b1x)} / (1 + e^{(b0+b1x)})$

On the right-hand side, divide the numerator and denominator by e(b0+b1x).
$p = 1 / (1 + e^{-(b0+b1x)})$

Likewise, for a logistic model with 'n' predictors, the equation is as follows:
p = 1/ (1 + e-(b0+b1x1+b2x2+b3x3+----+bnxn)

Multinomial Naive Bayes:

The Multinomial Naive Bayes method is a common Bayesian learning strategy (NLP). Using the Bayes theorem, the software estimates the tag of a text, such as an email or a newspaper piece. It assesses the likelihood of each tag for a given sample and returns the tag with the highest possibility. When the characteristics indicate frequency, the multinomial Naive Bayes algorithm is utilized.
Let's say you have a text document and you want to extract all the unique words and turn them into numerous features, each representing the number of words in the document. We have frequency as a feature in this situation. We utilize multinomial Naive Bayes in this case.
It disregards the absence of the characteristics. As a result, if the frequency is zero, the likelihood of that feature occurring is also zero, hence multinomial naive Bayes ignores that feature. It's well-known for helping with text categorization issues.
The multinomial Naive Bayes method has the following applications:

- Users may use this method to produce real-time forecasts since it is quick and efficient. For multi-class predictions, this approach is widely used. Using this approach, we can quickly determine the probability of numerous target classes.
- This algorithm is used by email providers (such as Gmail) to determine whether or not an email is spam. This

algorithm is fantastic at detecting spam.

- Its feature independence assumption, as well as its efficacy in addressing multi-class issues, makes it ideal for Sentiment Analysis. Sentiment Analysis is the process of determining if a target group's sentiments are good or negative (customers, audience, etc.)

Working:

The Naive Bayes method is a strong tool for analyzing text input and solving issues with numerous classes. Because the Naive Bayes theorem is based on the Bayes theorem, it is necessary to first comprehend the Bayes theorem notion.

The Bayes theorem, which was developed by Thomas Bayes, determines the likelihood of an action occurrence depending on existing knowledge of the event's circumstances. It's based on the formula below:

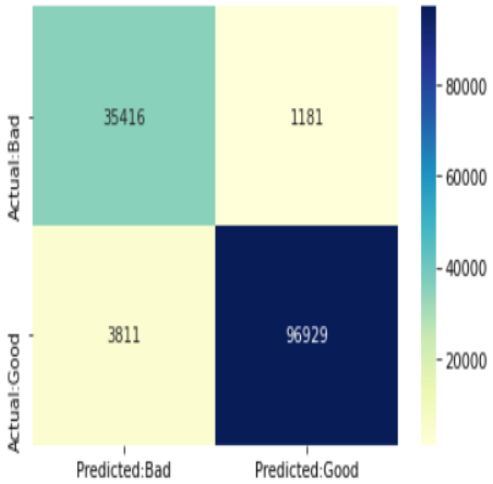
P(A|B) = P(A) * P(B|A)/P(B)

When predictor B is already available, we calculate the frequency of class A.
P(B) = prior probability of B
P(A) = prior probability of class A
P(B|A) =occurrence of predictor B given class A probability
This algorithm assists in determining the frequency of tags in the text.
Let's look at an example of the Naive Bayes method. We've collected a data set of weather conditions that includes bright, gloomy, and wet weather in the table below. Now we must anticipate whether or not the players will play based on the weather conditions.
-->By Comparing both the algorithms, we conclude Logistic regression is best for detecting malicious URLs

Result:

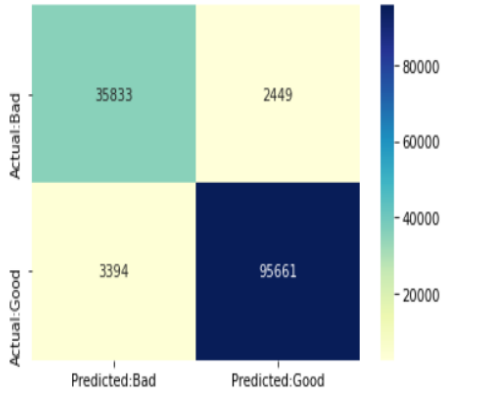
Logistic regression:

```
Training Accuracy : 0.9782480479795345
Testing Accuracy : 0.9636514559077306

CLASSIFICATION REPORT

              precision    recall  f1-score   support

         Bad       0.90      0.97      0.93     36597
        Good       0.99      0.96      0.97    100740

    accuracy                           0.96    137337
   macro avg       0.95      0.96      0.95    137337
weighted avg       0.97      0.96      0.96    137337


CONFUSION MATRIX
Out[46]: <matplotlib.axes._subplots.AxesSubplot at 0x1ec84c387c8>
```



Multinomial Naive Bayes:

```
Training Accuracy : 0.9741437687040817
Testing Accuracy : 0.9574550194048217

CLASSIFICATION REPORT

              precision    recall  f1-score   support

         Bad       0.91      0.94      0.92     38282
        Good       0.98      0.97      0.97     99055

    accuracy                           0.96    137337
   macro avg       0.94      0.95      0.95    137337
weighted avg       0.96      0.96      0.96    137337


CONFUSION MATRIX
Out[51]: <matplotlib.axes._subplots.AxesSubplot at 0x1ec84f9c908
```



Conclusion:

Logistic Regression is giving 96% accuracy compared to multinomial Naive Bayes Theorem, so here we can conclude that we can detect the malicious links by implementing the logistic regression method.

Colab link:

https://colab.research.google.com/drive/1EZxmflicAOdlt6w0t-wzaX3S0fCkMCDM?usp=sharing

Dataset link:

https://www.kaggle.com/datasets/taruntiwarihp/phishing-site-urls

References:

- https://datareportal.com/global-digital-overview#:~:text=There%20are%204.95%20billion%20internet,500%2C000%20new%20users%20each%20day.
- https://sectigo.com/resource-library/what-happens-when-i-visit-a-malicious-website#:~:text=Once%20you%20visit%20a%20page,scrapes%20personal%20information%20from%20you.
- https://cheapsslsecurity.com/blog/what-is-a-malicious-url/
- https://gatefy.com/blog/what-malicious-url/
- https://www.hindawi.com/journals/scn/2021/5518528/#data-availability
- https://www.cyber.gov.au/acsc/view-all-content/glossary/malicious-links
- https://machinelearningmastery.com/prepare-text-data-deep-learning-keras/
- https://towardsdatascience.com/vectorization-implementation-in-machine-learning-ca652920c55d