

**A Mini Project Report  
on  
MALICIOUS URL DETECTION USING MACHINE  
LEARNING**

**Submitted in Partial fulfillment of the requirements**

**for the award of degree of  
BACHELOR OF TECHNOLOGY**

**in**

**Information Technology**

**by**

***N. Prasanna Laxmi(19WH1A1283)***

***B . Navya (19WH1A12A2)***

***T . Sri Latha(20WH5A1207)***

***Ch. Madhuri (20WH5A1208)***

***Under the esteemed guidance of***

***Mr. K. Srikar Goud***

***Assistant Professor***



***Department of Information Technology***

***BVRIT HYDERABAD College of Engineering for Women***

***Rajiv Gandhi Nagar, Nizampet Road, Bachupally, Hyderabad – 500090***

***(Affiliated to Jawaharlal Nehru Technological University, Hyderabad)***

***(NAAC 'A' Grade & NBA Accredited- ECE, EEE, CSE IT)***

**January, 2023**

# DECLARATION

We hereby declare that the work presented in this project entitled “**MALICIOUS URL DETECTION USING MACHINE LEARNING**” submitted towards completion of Minor Project in IV year I sem of B.Tech IT at “BVRIT HYDERABAD College of Engineering for Women”, Hyderabad is an authentic record of our original work carried out under the esteemed guidance of **Mr. K. Srikar Goud , Assistant Professor**, Department of Information Technology.

**N. Prasanna Laxmi(19WH1A1283)**

**B . Navya (19WH1A12A2)**

**T . Sri Latha(20WH5A1207)**

**Ch. Madhuri (20WH5A1208)**



## **BVRIT HYDERABAD**

### **College of Engineering for Women**

**Rajiv Gandhi Nagar, Nizampet Road, Bachupally, Hyderabad – 500090**

**(Affiliated to Jawaharlal Nehru Technological University, Hyderabad)**

**(NAAC 'A' Grade & NBA Accredited- ECE, EEE, CSE IT)**

## **CERTIFICATE**

This is to certify that the minor-project report on **“MALICIOUS URL DETECTION USING MACHINE LEARNING”** is a bonafide work carried out by **N. Prasanna Laxmi(19WH1A1283), B . Navya (19WH1A12A2), T . Sri Latha(20WH5A1207)** and **Ch. Madhuri (20WH5A1208)** in the fulfillment for the award of B.Tech degree in **Information Technology, BVRIT HYDERABAD College of Engineering for Women, Bachupally, Hyderabad** affiliated to Jawaharlal Nehru Technological University, Hyderabad under my guidance and supervision.

The results embodied in the project work have not been submitted to any other university or institute for the award of any degree or diploma.

#### **Internal Guide**

**Mr. K. Srikar Goud**

**Assistant Professor**

**Department of IT**

#### **Head of the Department**

**Dr. Aruna Rao S L**

**Professor & HoD**

**Department of IT**

#### **External Examiner**

## ACKNOWLEDGEMENT

We would like to express our profound gratitude and thanks to **Dr. K. V. N. Sunitha, Principal, BVRIT HYDERABAD** for providing the working facilities in the college.

Our sincere thanks and gratitude to **Dr. Aruna Rao S L, Professor & Head, Department of IT, BVRIT HYDERABAD** for all the timely support, constant guidance and valuable suggestions during the period of our project.

We are extremely thankful and indebted to our internal guide, **Mr. K. Srikar Goud, Assistant Professor, Department of IT, BVRIT HYDERABAD** for his constant guidance, encouragement and moral support throughout the project.

Finally, we would also like to thank our Project Coordinators **Ms. K. S. Niraja, Assistant Professor, Mr. Ch. Anil Kumar, Assistant Professor**, all the faculty and staff of Department of IT who helped us directly or indirectly, parents and friends for their cooperation in completing the project work.

N. Prasanna Laxmi(19WH1A1283)

B . Navya (19WH1A12A2)

T . Sri Latha(20WH5A1207)

Ch. Madhuri (20WH5A1208)

# ABSTRACT

The World Wide Web supports a wide range of criminal activities such as spam-advertised e-commerce, financial fraud, and malware dissemination. Although the precise motivations behind these schemes may differ, the common denominator lies in the fact that unsuspecting users visit their sites. These visits can be driven by email, web search results, or links from other web pages. In all cases, however, the user is required to take some action, such as clicking on a desired Uniform Resource Locator (URL). To identify these malicious sites, the web security community has developed blacklisting services. Inevitably, many malicious sites are not blacklisted either because they are too recent or were never or incorrectly evaluated. The proposed detection system consists of a new set of URLs features and behaviours, a machine learning algorithm. The experimental results show that the proposed URL attributes and behaviour can help improve the ability to detect malicious URL significantly. This is suggested that the proposed system may be considered as an optimized and friendly used solution for malicious URL detection. In this work we study the performance of several well-known classifiers, namely Naïve Bayes, Support Vector Machines, Multi-Layer Perceptron, Decision Trees, Random Forest, and k-Nearest Neighbours. We took a publicly available dataset which is labelled with legitimate and malicious URLs. By using classification algorithms we detect whether the given URL is malicious or safe.

## LIST OF FIGURES

Figure No.	Figure Name	Page No.
1.1.1	Malicious websites	2
3.2.1	Architecture Design	7
3.3.1	Usecase Diagram	9
3.4.1	Flow Diagram	10
4.4.1	Output after loading dataset	17
4.4.2	Accuracy of the model	18
4.4.3	Prediction of URLs	19
4.4.4	Another Prediction of URLs	20

# TABLE OF CONTENTS

TOPIC	PAGE NO.
<b>Abstract</b>	<b>V</b>
<b>List of Figures</b>	<b>VI</b>
<b>1. Introduction</b>	<b>1</b>
<b>1.1 Objective</b>	
<b>1.2 Problem Definition</b>	
<b>1.3 Aim of the Project</b>	
<b>2. Literature Survey</b>	<b>4</b>
<b>2.1 Major Issues</b>	
<b>3. System Analysis and Design</b>	<b>6</b>
<b>3.1 Proposed System</b>	
<b>3.2 Architecture Design</b>	
<b>3.3 Usecase Diagram</b>	
<b>3.4 Project Flow</b>	
<b>4. Implementation</b>	<b>12</b>
<b>4.1 Module Names</b>	
<b>4.2 Designing and Training of the Model</b>	
<b>4.3 Testing and Prediction</b>	
<b>4.4 Results</b>	
<b>5. Conclusion and Future Scope</b>	<b>22</b>
<b>References</b>	<b>23</b>

# 1. Introduction

## 1.1 Objective

A website link that is intended to spread viruses, phishing scams, and other illegal actions is known as a malicious URL. A user can download computer viruses including trojan horses, ransomware, worms, and malware by clicking a malicious URL. These infections ultimate objectives include gaining access to personal data, harming the user's device, and generating revenue. They might also ruin the network of the business, causing losses.

New communication technologies have significantly impacted business development and promotion across a wide range of applications. The World Wide Web has becoming increasingly significant. Unfortunately, new sophisticated methods for attacking and defrauding users are brought on by technological improvements. These assaults can involve installing malware on a user's computer or using rogue websites to sell counterfeit goods or divulge sensitive information that can be used to steal money and commit financial fraud. Assaults can take many different forms, including direct hacking attempts, drive-by exploitsphishing, watering holes, social engineering, man-in-the-middle attacks, loss or theft services, etc. There are many different types of attacks, and new ones are developed daily. It is challenging. A malicious URL can also be employed to entice users to enter their personal data on a bogus website. These folks are compelled to divulge their private information to strangers as a result. They make use of the data for a covert purpose. These rogue URLs have the potential to do a lot of damage.

Spam, Drive-by Download, Social Engineering, and Phishing are common forms of assaults that use malicious URLs. Spam is the term used to describe forced transmission of unsolicited messages including advertising or phishing that we did not ask for and do not want to receive. These assaults have done a great deal of harm. Drive-by downloads are when malware is downloaded while accessing a URL. Finally, by pretending to be legitimate web pages, social engineering and phishing assaults trick users into disclosing private and sensitive information. By embedding several pieces of malicious code in the HTML code of the altered web site, the attackers make clones of widely used user web pages like Facebook and Google and infect victim machines.



The goal of this work is to use feature extraction and the logistic regression algorithm to determine whether the provided url is malicious or not. to keep out numerous cyber security risks.



**Figure 1.1.1:** malicious websites

## 1.2 Problem Definition

A website link that is intended to spread viruses, phishing scams, and other illegal actions is known as a malicious URL. A user can download computer viruses including trojan horses, ransomware, worms, and malware by clicking a malicious URL. A malicious URL can also be employed to entice users to enter their personal data on a malicious website. These folks are compelled to divulge their private information to strangers as a result. They make use of the data for a covert purpose. These rogue URLs have the potential to do a lot of damage. we will build a machine learning model that can be able to detect these malicious URLs. We will train our model using a dataset with URLs labeled both bad and good.

## 1.3 Aim of the project

Increase in cybersecurity attacks such as ransomware, phishing, injection of malware, etc. on different websites all over the world. As a result of this, various financial institutions, e-commerce companies, and individuals incurred huge financial losses. In such type of scenario containing a cyber security attack is a major challenge for cyber security professionals as different types of new attacks are coming day by day. The aim of our project is we address the detection of malicious URLs as a multi-class classification problem. we classify the raw URLs into different class types such as benign or safe URLs.

## 2. Literature Survey

### 2.1 Related Work

- The authors Mohammed Nazim Feroz and, Susan Mengel has describes an approach that classifies URLs automatically based on their lexical and host-based features. These methods are able to learn highly analytical models by extracting and automatically Mahout is established for such scalable machine learning problems, and online learning is considered over batch learning. The classifier achieves 93-95 % accuracy by detecting a large number of phishing hosts, while maintaining a modest false positive rate.
- Justin Ma, Lawrence K. Saul, Stefan Savag and, Geoffrey M. Voelker describes an approach to this problem based on automated URL classification, using statistical methods to discover the tell-tale lexical and host-based properties of malicious Web site URLs. These methods are able to learn highly analytical models by extracting and repeatedly examining tens of thousands of features potentially indicative of suspicious URLs. The resulting classifiers obtain 91-94% accuracy, detecting large numbers of malicious Web sites from their URLs, with only modest false positives. .
- Frank Vanhoenshoven, Gonzalo N apoles, Rafael Falcot, Koen Vanhoof and Mario K"oppent Universiteit Hasselt Campus Diepenbeek determines online learning approaches for detecting malicious Web sites (those involved in criminal scams) using lexical and host-based features of the related URLs. We show that this application is mostly suitable for online algorithms as the size of the training data is larger which can be efficiently processed in batch also the distribution of features that typify malicious URLs is changing unceasingly.
- Singh et al. propose a method for malware detection by applying the Support Vector Machine. Kazemian et al. have used machine learning techniques such as K-Nearest Neighbor, Support Vector Machines, Naive Bayes Classifier, and K Means rather than tradi-tional methods of detecting whether they exist in a predetermined blacklist for detecting harmful webpages.

- Canali et al. proposed a model that applied Naive Bayes for Malicious URL Detection. The extreme Learning Machines (ELM) approach is used for classifying the phishing websites .

## **2.2 Major Issues**

One of the primary flaws of the Blacklist approach is its inability to identify newly created harmful URLs. Additionally, several typical attacks are identified using a heuristic manner. It is a more sophisticated Blacklist approach strategy.

### **3. System Analysis and Design**

#### **3.1 Proposed System**

The proposed malicious URL detection system using machine learning is used to detect whether the given URL is safe or malicious . A list of URLs that have been categorised as malicious or helpful, characterising each URL using variety of characteristics like the number of dots present in the URL, and its location. The algorithm we used is Logistic regression it is one of the most popular algorithms for binary classification. Given a set of examples with features, the goal of logistic regression is to output values between 0 and 1, which can be interpreted as the probabilities of each example belonging to a particular class. Binary classification technique, also known as binary regression technique, is used in this model to train it. Benefit of suggested approach In comparison to other machine learning algorithms is the suggested method achieves the highest learning accuracy.

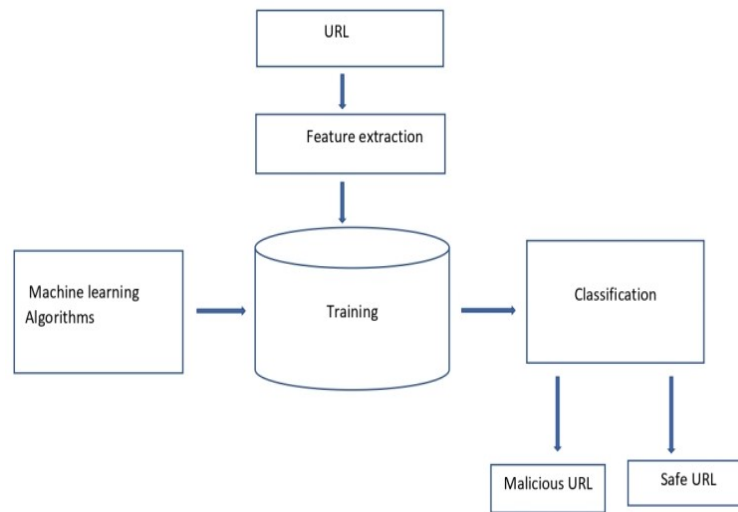
#### **3.2 Architecture Design**

The four main phases which we undergo for developing the application “Malicious URL Detection Using Machine Learning” is:

- 1)Data Collection
- 2)Preprocessing
- 3)Designing and Training of the CNN and RNN model
- 4)Prediction

We will utilise a dataset with URLs tagged as both bad and good in order to construct this model. The model will collect meaningful knowledge from the dataset and utilise that knowledge to create predictions. There are two columns for the dataset. The real URL linkages are represented by the first column, url. Both undesirable and desirable URLs are included in the second column, label. Eliminating noise from our dataset is known as dataset cleaning.

Noise is made up of repeating words, needless punctuation, and text data. The model will be able to concentrate solely on the most crucial variables in the dataset once the noise has been eliminated from the dataset.



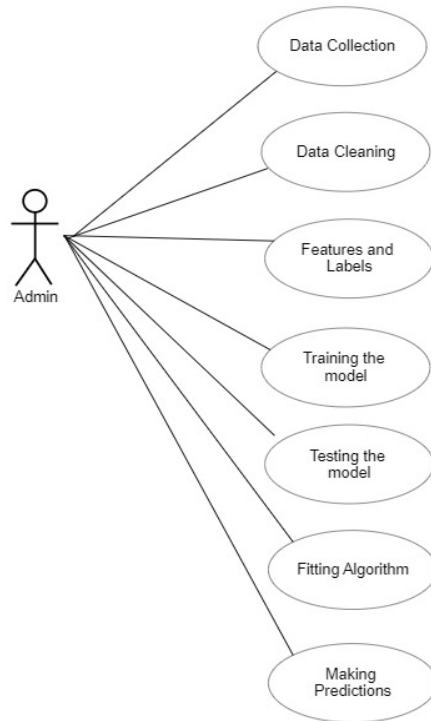
**Figure 3.2.1:** Architecture Design

The model performance will improve as a result. Predictions made by the model will be precise. We now need to add features and labels to our dataset after this text cleaning.

### 3.3 Usecase diagram:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. In Unified Modeling Language (UML), systems are presented at different levels of detail to show a specific perspective in the system's design. Use case diagrams are considered UML diagrams. A use case diagram is a visual representation of the different ways and possible scenarios of using a system. It illustrates how a user will perform actions and interact with a particular system, such as a website or an app. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

The main purpose of a use case diagram is to portray the dynamic aspect of a system. It accumulates the system's requirement, which includes both internal as well as external influences. It invokes persons, use cases, and several things that invoke the actors and elements accountable for the implementation of use case diagrams. It represents how an entity from the external environment can interact with a part of the system. It is essential to analyze the whole system before starting with drawing a use case diagram, and then the system's functionalities are found. And once every single functionality is identified, they are then transformed into the use cases to be used in the use case diagram.



**Figure 3.3.1:** Usecase Diagram

### 3.4 project flow:

Steps involved in Detecting Malicious URL are :

#### 1. Data Collection :

we will use a dataset with URLs labeled both bad and good. The model will learn from the dataset and gain useful knowledge which it will use to make predictions.

#### 2. Feature Extraction :

Features are the unique data points in our dataset that are used as input for the model during training. Features are represented by the url column, which is our input column.

#### 3. Training and Testing the model :

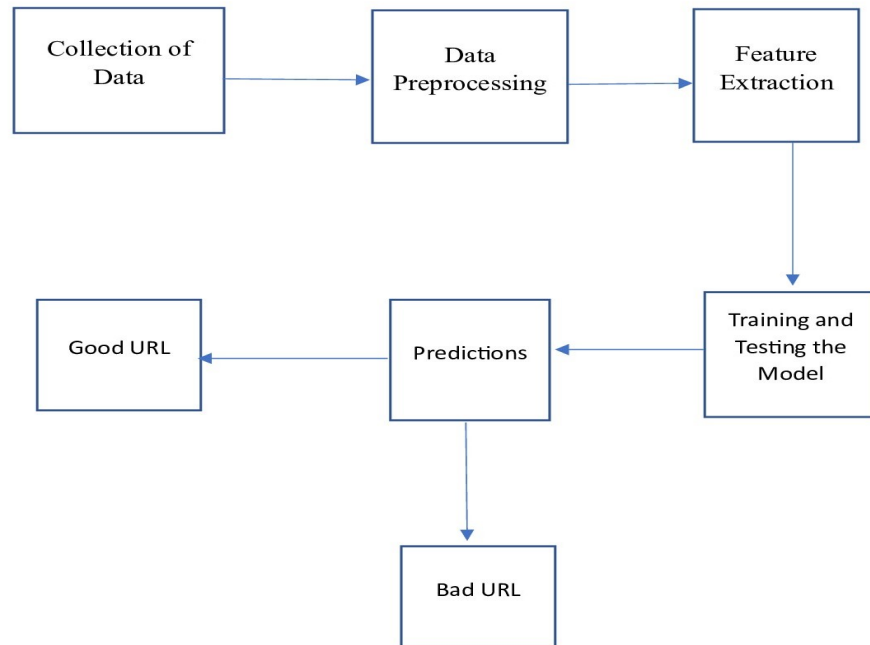
We divided our dataset into a train and test by using some machine learning algorithms the



model will be trained and that was used to feed our model and fit algorithm. we used Logistic regression algorithm, a probabilistic statistical method, is a popular supervised machine learning algorithm used for classification and optimization problems. The algorithm has shown great performance for a variety of common applications such as email spam detection, diabetes prediction, cancer detection, etc. In logistic regression, the sigmoid function (also called the logistic function) and a threshold are used to calculate the likelihood of a label. Logistic regression estimates the likelihood of each label for the test sample and is typically deployed for predicting the target value with categorical dependent variables, for example, with binary labels ('true' or 'false', 'yes' or 'no').

### **5. Prediction :**

Prediction refers to the output of an algorithm after it has been trained on a historical dataset and applied to new data when forecasting the likelihood of a particular outcome. The algorithm will generate probable values for an unknown variable for each record in the new data, allowing the model builder to identify what that value will most likely be. By using the algorithm the model will predict the given URL whether is Good or bad.



**Figure 3.4.1:** Flow Diagram

## 4. Implementation

### 4.1 Module Names

1. Data Collection
2. Data Preprocessing
3. Feature Extraction
4. Training the model
5. Predictions

#### 1. Data Collection

The dataset contains two columns the first column represents the URL and second column represents the label which contains both good and bad URLs. The next step is cleaning of our dataset.

#### 2. Data Preprocessing

Our dataset has to be cleaned by taking out the noise. Noise in text data includes unnecessary characters, punctuation, and words that are used repeatedly. The model will be able to focus purely on the most crucial variables in the dataset if the dataset's noise is removed. The performance of the model will rise as a result. The model will be able to anticipate outcomes correctly. To remove repetitions from the dataset, we first separate the texts. We will finally take the com out of each URL. To clean up our dataset, we will develop a unique Python method.

#### 2. Feature Extraction

To divide our text into dash, dot, and slash segments, we utilise the makeTokens method. This will guarantee that the text is free of these extra characters. The tool will also eliminate words from the text that are unnecessary. The total Tokens.delete('com') method is then used to remove the term 'com'. A clean text will then be returned by the function. We now need to add features and labels to our dataset after this text cleaning. The features in our dataset are the distinct data points that are used as input for the model during training. The url column, which is our input column, represents features. a label is the model's output after prediction. It is re-

presented using the label column. The model's output can either be bad or good.

To build our machine learning model, we have various Python packages that are essential for this process. We will import all the important packages. We will use logistic algorithm to train our model. This algorithm will enable our model to understand patterns and relationships in our dataset. The model will gain useful knowledge and insight, which it will use to make predictions.

### **Training and Testing**

#### **train\_test\_split:**

This is the function in Sklearn model selection for splitting data arrays into two subsets: for training data and for testing data.

#### **TfidfVectorizer:**

This package will enable the model to understand and manipulate text data. Text is a big problem for machines, machines cannot consume text in its raw form. We need to convert text into vectors of numbers that machines can read and understand. TfidfVectorizer is used to convert the raw text data into vectors of numbers that represents the original text. This text is converted based on the frequency of occurrence of each word in the text data.

#### **Convert the text data into vectors of numbers:**

We convert the text using TfidfVectorizer and also pass makeTokens as a parameter. makeTokens is the function used to clean our text.

#### **Dataset splitting:**

We will use train\_test\_split to split our dataset into two sets. One set will be used for model training and the other one will be used for model testing. The ratio will ensure that 80 % of the dataset is used to train the model and 20 % will be used to test the model.

**Model building using LogisticRegression:**

We will initialize the LogisticRegression algorithm . After initializing the algorithm, we will fit the algorithm onto our training dataset. The accuracy score is 96.164%. This is a very high accuracy score and implies that our model was well trained. The model learned a lot from the dataset during the training phase and can now be used to make predictions. We will use several URLs and see if the model can classify if the URL is bad or good.

## 4.2 Designing and Training of the model:

```
# Loading dataset
import pandas as pd
urls_data = pd.read_csv("urldata.csv")
urls_data.head()

# Dataset cleaning
def makeTokens(f):
    tkns_BySlash = str(f.encode('utf-8')).split('/')

# make tokens after splitting by slash
total_Tokens = []
for i in tkns_BySlash:
    tokens = str(i).split('-')

# make tokens after splitting by dash
tkns_ByDot = []
for j in range(0,len(tokens)):
    temp_Tokens = str(tokens[j]).split('.')

# make tokens after splitting by dot
tkns_ByDot = tkns_ByDot + temp_Tokens
total_Tokens = total_Tokens + tokens + tkns_ByDot
total_Tokens = list(set(total_Tokens))
```

```
#remove redundant tokens

if 'com' in total_Tokens:
    total_Tokens.remove('com')

# removing .com since it occurs a lot of times and it should not be included in our feature
return total_Tokens

# Features and labels
url_list = urls_data[\"url\"]
y = urls_data[\"label\"]

# Importing packages
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer

# Convert the text data into vectors of numbers
vectorizer = TfidfVectorizer(tokenizer=makeTokens)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Model building using LogisticRegression
logit = LogisticRegression()
logit.fit(X_train, y_train)

# Calculating the model's accuracy score
print('Accuracy ',logit.score(X_test, y_test))

Accuracy 0.96163771063
```

### 4.3 Testing and Prediction:

To make predictions, we will use several URLs and see if the model can classify if the URL is bad or good.

```
X_predict = ["https://www.section.io/engineering-education/",  
"https://www.youtube.com/",  
"https://www.traversymedia.com/",  
"https://www.kleinhundezuhause.com",  
"https://www.mecymiafinance.com",  
"https://www.atlanticoceanicoilandgas.com"]
```

```
X_predict = vectorizer.transform(X_predict)  
New_predict = logit.predict(X_predict)  
print(New_predict)
```

# Another Prediction

```
X_predict1 = ["www.buyfakebillsonlinee.blogspot.com",  
"www.unitedairlineslogistics.com",  
"www.stonehousedelivery.com",  
"www.silkroadmeds-onlinepharmacy.com"]
```

# To make predictions

```
X_predict1 = vectorizer.transform(X_predict1)  
New_predict1 = logit.predict(X_predict1)  
print(New_predict1)
```



## 4.4 Results:

```
[1] # EDA Packages
import pandas as pd

[3] # Load Url Data
urls_data = pd.read_csv("urldata.csv")

[4] type(urls_data)

pandas.core.frame.DataFrame

urls_data.head()
```

	url	label
0	diaryofagameaddict.com	bad
1	espdesign.com.au	bad
2	iamagameaddict.com	bad
3	kalantzis.net	bad
4	slightlyoffcenter.net	bad

**Figure 4.4.1:** Output after loading dataset

```
✓ [12] # Accuracy of Our Model
0s print("Accuracy ",logit.score(X_test, y_test))

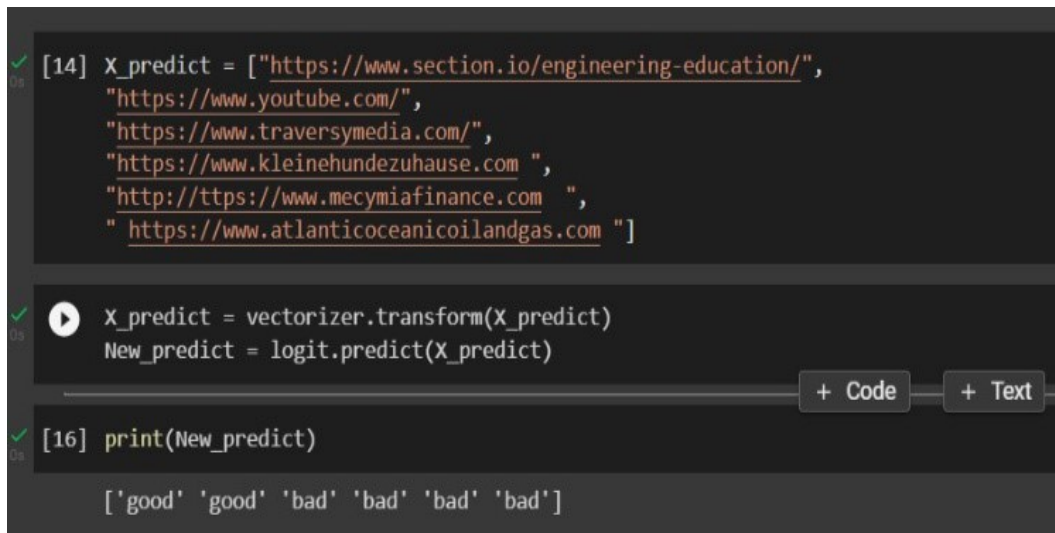
Accuracy 0.9617447349957785

▼ Predicting With Our Model

✓ [13] X_predict = ["https://www.section.io/engineering-education/",
0s "https://www.youtube.com/",
"https://www.traversymedia.com/",
"https://www.kleinhundezuhause.com ",
"http://https://www.mecymiafinance.com ",
" https://www.atlanticoceanicoilandgas.com "]

...ll.k.k.
```

**Figure 4.4.2 :** Accuracy of the model (96.2 %)



A screenshot of a Jupyter Notebook interface with a dark theme. It shows three code cells. The first cell (index 14) defines a list of URLs. The second cell (index 15) contains a play button icon and code to transform the URLs and make predictions. The third cell (index 16) prints the results. The output of the third cell is displayed below the code.

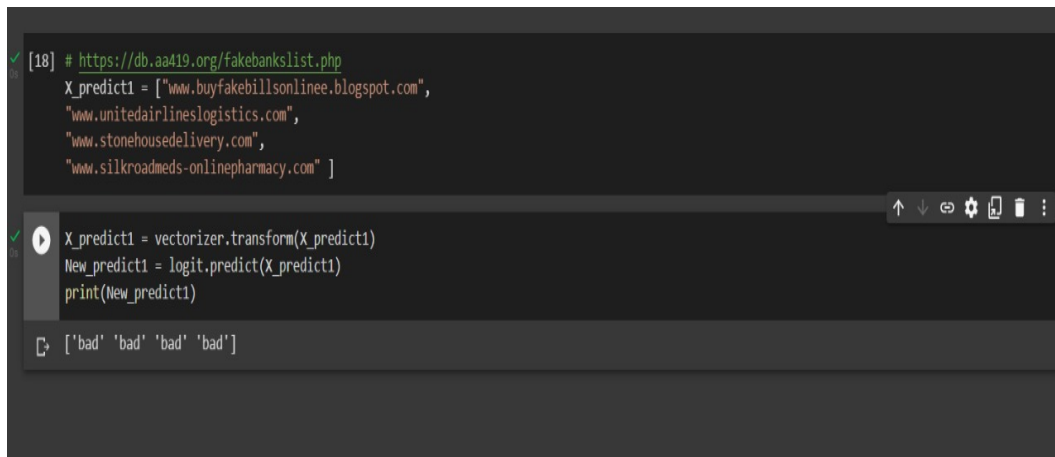
```
[14] x_predict = ["https://www.section.io/engineering-education/",  
                "https://www.youtube.com/",  
                "https://www.traversymedia.com/",  
                "https://www.kleinhundezuhause.com ",  
                "http://https://www.mecymiafinance.com ",  
                " https://www.atlanticoceanicoilandgas.com "]
```

```
[15] X_predict = vectorizer.transform(X_predict)  
     New_predict = logit.predict(X_predict)
```

```
[16] print(New_predict)
```

['good' 'good' 'bad' 'bad' 'bad' 'bad']

**Figure 4.4.3 :** Prediction of URLs



A screenshot of a Jupyter Notebook interface. The top cell contains a comment and a list of URLs. The bottom cell contains code to transform the URLs and predict their status. The output of the bottom cell is displayed below the code.

```
[18] # https://db.aa419.org/fakebankslist.php
X_predict1 = ["www.buyfakebillsonline.blogspot.com",
              "www.unitedairlineslogistics.com",
              "www.stonehousedelivery.com",
              "www.silkroadmeds-onlinepharmacy.com" ]

X_predict1 = vectorizer.transform(X_predict1)
New_predict1 = logit.predict(X_predict1)
print(New_predict1)
```

```
['bad' 'bad' 'bad' 'bad']
```

**Figure 4.4.4 :** Another Prediction of URLs

## 5. Conclusion and Future Scope

Malicious URL detection plays a serious role for many cyber security applications, and networking applications. The majority of computer attacks are launched by visiting a malicious webpage. In this we showed a broad and organized study on malicious URL detection using logistic regression algorithm. We also identify the requirements for detecting malicious URLs. As a result, our proposed system is used to determine the malicious and legitimate URLs as a service of real-world cyber security applications.

In the future, phishing detection will be considerably faster than with any other technique if we have access to structured datasets of phishing. In the future, we can combine any other two or more classifiers to achieve the highest accuracy. Additionally, in order to enhance the system's speed, we intend to investigate numerous phishing strategies that make advantage of HTML and JavaScript capabilities, Network-based features, Content-based features, and Lexical and Network-based aspects of web pages. We specifically extract information from URLs and subject them to different classifiers.

## References

- [1] Do Xuan, C., Nguyen, H.D., Nikolaevich, T.V., et al.: Malicious url detection based on machine learning. *Int. J. Adv. Comput. Sci. Appl.* 11 (2020).
- [2] Johnson, C., Khadka, B., Basnet, R.B., Doleck, T.: Towards detecting and classifying malicious urls using deep learning. *J. Wirel. Mob. Networks Ubiquitous Comput. Dependable Appl.* 11, 31–48 (2020).
- [3] Li, T., Kou, G., Peng, Y.: Improving malicious urls detection via feature engineering: linear and nonlinear space transformation methods. *Inf. Syst.* 91, 101494 (2020).
- [4] Tung, S.P., Wong, K.Y., Kuzminykh, I., Bakhshi, T., Ghita, B.: Using a machine learning model for malicious url type detection. In: *Internet of Things, Smart Spaces, and Next Generation Networks and Systems*, pp. 493–505 (2021).
- [5] Bell, S., Komisarczuk, P.: An analysis of phishing blacklists: google safe browsing, openphish, and phishtank. In: *Proceedings of 1st Australasian Computer Science Week Multiconference (ACSW'16)*, pp. 1–11 (2020).
- [6] Doyen Sahoo, Chenghao lua, Steven C. H. Hoi, Malicious URL Detection using Machine Learning: A Survey, arXiv:1701.07179v3 [cs.LG], 21 Aug 2019.
- [7] Kim, D.: Potential risk analysis method for malware distribution networks. *IEEE Access* 7, 185157–185167 (2019).
- [8] Y.-C. Chen, Y.-W. Ma and J.-L. Chen, "Intelligent malicious url detection with feature analysis", 2020 IEEE Symposium on Computers and Communications (ISCC), pp. 1-5, 2020.