

PHISHING URL DETECTION USING MACHINE LEARNING

*A main Project Report submitted in the partial fulfillment of the
requirements for the award of the degree*

BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING

Submitted by

| | |
|------------------|--------------|
| Sk. Chan basha | (19471A05O5) |
| Ch. Sreekanth | (19471A05K4) |
| M. Naga srikanth | (20475A0513) |

Under the esteemed guidance of

G.Saranya MTech.



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

NARASARAOPETA ENGINEERING COLLEGE:

NARASARAOPET (AUTONOMOUS)

(Affiliated to J.N.T.U.Kakinada ,Approved by AICTE&Acredited by NBA)

2022-2023

**NARASARAOPETA ENGINEERING COLLEGE: NARASARAOPETA
(AUTONOMOUS)**

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



CERTIFICATE

This is to certify that the main project entitled **”PHISHING URL DETECTION
USING MACHINE LEARNING”** is a bonafide work done by **“Sk.Chan
basha(19471A05O5), Ch.Sreekanth(19471A05K4), M.Naga Srikanth(20475A0513)”**
in partial fulfilment of the requirements for the award of the degree of BACHELOR
OF TECHNOLOGY in the department of COMPUTER SCIENCE AND
ENGINEERING during 2022-2023.

PROJECT GUIDE

G.Saranya ,MTech.

PROJECT CO-ORDINATOR

M.Sireesha ,MTech.,(Ph.D)

HEAD OF THE DEPARTMENT

Dr.S.N.TirumalaRao,M.Tech.,Ph.D

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

We wish to express our thanks to carious personalities who are responsible for the completion of the project. We are extremely thankful to our beloved chairman sir **M.V. Koteswara Rao, B.sc** who took keen interest in us in every effort throughout this course. We owe out gratitude to our principal **Dr.M.Sreenivasa Kumar, M.Tech,Ph.D.,** for his kind attention and valuable guidance throughout the course.

We express our deep felt gratitude to **Dr.S.N.Tirumala Rao, M.Tech, Ph.D** Professor and H.O.D. CSE department and our guide **G.Saranya,M.Tech.** of CSE department whose valuable guidance and unstinting encouragement enable us to accomplish our project successfully in time.

We extend our sincere thanks to **M.Sireesha B.Tech., M.Tech, (Ph.D)** Associate Professor coordinator of the project for extending her encouragement. Their profound knowledge and willingness have been a constant source of inspiration for us throughout this project work.

We extend our sincere thanks to all other teaching and non-teaching staff to department for their cooperation and encouragement during our B.Tech degree. We have no words to acknowledge the warm affection, constant inspiration and encouragement that we receive from our parents. We affectionately acknowledge the encouragement received from our friends those who involved in giving valuable suggestions had clarifying out all doubts which had really helped us in successfully completing our project.

By

Sk. Chan basha (19471A05O5)

Ch. Sreekanth (19471A05K4)

M. Naga Srikanth (20475A0513)

ABSTRACT

In recent times, there is a massive increase in the number of devices that are being connected to the internet. These devices include but are not limited to smartphones, IoT, and cloud networks. In comparison to other possible cyber-attacks, these days, hackers are targeting these devices with phishing attacks since it exploits human vulnerabilities rather than system vulnerabilities. In a phishing attack, an online user is deceived by a seemingly trusted entity to give their personal data, i.e., login credentials or credit card details.

When this private information is leaked to the hackers, this information becomes the source of other sophisticated attacks. In recent times many researchers have proposed the machine learning-based approach to solve phishing attacks; however, they have used a large number of features to develop reliable phishing detection techniques. A large number of features requires large processing powers to detect phishing, which makes it very much unsuitable for resource constrained devices. Our proposed model addresses this issue and considers only needs nine lexical features for effectively detecting phishing attacks. ISCXURL-2016 dataset is used for our experimental purpose, where 11964 instances of legitimate and phishing URLs are present. Our proposed was tested approach against different machine learning classifiers and have obtained the highest accuracy of 97.61% with the Random forest algorithm.



INSTITUTE VISION AND MISSION

INSTITUTION VISION

To emerge as a Centre of excellence in technical education with a blend of effective student centric teaching learning practices as well as research for the transformation of lives and community,

INSTITUTION MISSION

M1: Provide the best class infra-structure to explore the field of engineering and research

M2: Build a passionate and a determined team of faculty with student centric teaching,imbibing experiential, innovative skills

M3: Imbibe lifelong learning skills, entrepreneurial skills and ethical values in students for addressing societal problems





DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VISION OF THE DEPARTMENT

To become a centre of excellence in nurturing the quality Computer Science & Engineering professionals embedded with software knowledge, aptitude for research and ethical values to cater to the needs of industry and society.

MISSION OF THE DEPARTMENT

The department of Computer Science and Engineering is committed to

M1: Mould the students to become Software Professionals, Researchers and Entrepreneurs by providing advanced laboratories.

M2: Impart high quality professional training to get expertize in modern software tools and technologies to cater to the real time requirements of the Industry.

M3: Inculcate team work and lifelong learning among students with a sense of societal and ethical responsibilities.

Program Specific Outcomes (PSO's)

PSO1: Apply mathematical and scientific skills in numerous areas of Computer Science and Engineering to design and develop software-based systems.

PSO2: Acquaint module knowledge on emerging trends of the modern era in Computer Science and Engineering

PSO3: Promote novel applications that meet the needs of entrepreneur, environmental and social issues.



Program Educational Objectives (PEO's)

The graduates of the programme are able to:

PEO1: Apply the knowledge of Mathematics, Science and Engineering fundamentals to identify and solve Computer Science and Engineering problems.

PEO2: Use various software tools and technologies to solve problems related to academia, industry and society.

PEO3: Work with ethical and moral values in the multi-disciplinary teams and can communicate effectively among team members with continuous learning.

PEO4: Pursue higher studies and develop their career in software industry.

Program Outcomes

- 1. Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
- 2. Problem analysis:** Identify, formulate, research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
- 3. Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
- 4. Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
- 5. Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
- 6. The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
- 7. Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

8. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

10. Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Project Course Outcomes (CO'S):

CO425.1: Analyse the System of Examinations and identify the problem.

CO425.2: Identify and classify the requirements.

CO425.3: Review the Related Literature

CO425.4: Design and Modularize the project

CO425.5: Construct, Integrate, Test and Implement the Project.

CO425.6: Prepare the project Documentation and present the Report using appropriate method.

Course Outcomes – Program Outcomes mapping

| | PO1 | PO2 | PO3 | PO4 | PO5 | PO6 | PO7 | PO8 | PO9 | PO10 | PO11 | PO12 | PSO1 | PSO2 | PSO3 |
|---------------|-----|-----|-----|-----|-----|-----|-----|-----|-----|------|------|------|------|------|------|
| C425.1 | | √ | | | | | | | | | | | √ | | |
| C425.2 | √ | | √ | | √ | | | | | | | | √ | | |
| C425.3 | | | | √ | | √ | √ | √ | | | | | √ | | |
| C425.4 | | | √ | | | √ | √ | √ | | | | | √ | √ | |
| C425.5 | | | | | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ |
| C425.6 | | | | | | | | | √ | √ | √ | | √ | √ | |

Course Outcomes – Program Outcome correlation

| | PO1 | PO2 | PO3 | PO4 | PO5 | PO6 | PO7 | PO8 | PO9 | PO10 | PO11 | PO12 | PSO1 | PSO2 | PSO3 |
|---------------|-----|-----|-----|-----|-----|-----|-----|-----|-----|------|------|------|------|------|------|
| C425.1 | 2 | 3 | | | | | | | | | | | 2 | | |
| C425.2 | | | 2 | | 3 | | | | | | | | 2 | | |
| C425.3 | | | | 2 | | 2 | 3 | 3 | | | | | 2 | | |
| C425.4 | | | 2 | | | 1 | 1 | 2 | | | | | 3 | 2 | |
| C425.5 | | | | | 3 | 3 | 3 | 2 | 3 | 2 | 2 | 1 | 3 | 2 | 1 |
| C425.6 | | | | | | | | | 3 | 2 | 1 | | 2 | 3 | |

Project mapping with various courses of Curriculum with Attained PO's:

| Name of the course from which principles are applied in this project | Description of the device | Attained PO |
|---|---|--------------------|
| C3.2.4, C3.2.5 | Gathering the requirements and defining the problem, plan to develop a smart bottle for health care using sensors. | PO1, PO3 |
| CC4.2.5 | Each and every requirement is critically analyzed, the process model is identified and divided into five modules | PO2, PO3 |
| CC4.2.5 | Logical design is done by using the unified modelling language which involves individual team work | PO3, PO5, PO9 |
| CC4.2.5 | Each and every module is tested, integrated, and evaluated in our project | PO1, PO5 |
| CC4.2.5 | Documentation is done by all our four members in the form of a group | PO10 |
| CC4.2.5 | Each and every phase of the work in group is presented periodically | PO10, PO11 |
| CC4.2.5 | Implementation is done and the project will be handled by the hospital management and in future updates in our project can be done based on air bubbles occurring in liquid in saline. | PO4, PO7 |
| CC4.2.8 CC4.2. | The physical design includes hardware components like sensors, gsm module, software and Arduino. | PO5, PO6 |

INDEX

| S.NO | CONTENTS | PAGE NO |
|-----------|--|-----------|
| 1. | Introduction | 1 |
| | 1.1 Application | 1 |
| | 1.2 Literature survey | 1 |
| | 1.3 Dataset and its preprocessing | 2 |
| | 1.4 Drawbacks | 3 |
| | 1.5 Objectives | 4 |
| | 1.6 Scope Of The Work | 6 |
| 2. | EXISTING METHODS | 7 |
| | 2.1 ML based phishing detection from URLs | 7 |
| | 2.2 Methodology | 7 |
| | 2.3 Results | 11 |
| | 2.4 conclusion | 12 |
| 3. | DETECTION OF PHISHING WEBSITES USING ML | 13 |
| | 3.1 Introduction | 13 |
| | 3.2 Methodology | 13 |
| | 3.3 Results | 15 |
| | 3.4 Conclusion | 16 |
| 4. | PROPOSED METHODS | 17 |
| | 4.1 Introduction | 17 |
| | 4.2 Methodology | 18 |
| | 4.3 Features extraction algorithm | 19 |
| | 4.4 Experimental details | 26 |
| | 4.5 Results and conclusion | 26 |
| 5. | IMPLEMENTATION | 31 |

| | | |
|------------|--|-----------|
| 6. | SCREENSHOTS | 38 |
| 7. | Experimental Results and Evaluation | 41 |
| 8. | CONCLUSION | 42 |
| 9. | REFERENCES | 43 |
| 10. | CONFERENCE PAPER | 44 |
| 11. | SIMILARITY CHECK REPORT | 49 |

LIST OF FIGURES

| | |
|---|----|
| 1. Execution of the data preprocessing module | 8 |
| 2. Execution of maliciousness analysis module | 9 |
| 3. Execution of word decomposer module | 10 |
| 4. Flowchart of proposed system | 14 |
| 5. Linear Regression plot of original o/p versus predicted o/p | 15 |
| 6. Machine Learning accuracy bar plot | 15 |
| 7. Proposed phishing detection approach | 18 |
| 8. Result of SVM classifier before and after hyper-parameter tuning | 19 |

LIST OF TABLES

| | |
|---|----|
| Table 1.1 : Distribution of data in dataset | 4 |
| Table 2.1 : Test Results of the classification algorithms | 11 |
| Table 2.2 : Confusion Matrix Results | 16 |
| Table 3.1 : Performance Measure uses in proposed approach | 28 |
| Table 3.2 : Results from different classifiers | 29 |
| Table 3.3 : Result of true and false positive classification | 29 |
| Table 3.4 : Results of different approaches available in literature | 30 |

Chapter 1

Introduction

In recent times, we can see a massive increase in the number of devices that are being connected to the internet. These devices include but are not limited to smartphones, IoT, and cloud networks. In comparison to other possible cyber-attacks, these days, hackers are targeting these devices with phishing attacks since it exploits human vulnerabilities rather than system vulnerabilities. In a phishing attack, an online user is deceived by a seemingly trusted entity to give their personal data, i.e., login credentials or credit card details. When this private information is leaked to the hackers, this information becomes the source of other sophisticated attacks. In recent times many researchers have proposed the machine learning-based approach to solve phishing attacks; however, they have used a large number of features to develop reliable phishing detection techniques. A large number of features requires large processing powers to detect phishing, which makes it very much unsuitable for resource constrained devices. Our approach addresses this issue that only needs nine lexical features for effectively detecting phishing attacks. ISCXURL-2016 dataset is used for our experimental purpose.

Applications

1. Phishing detection system can analyze any URL in real time in order to identify potential phishing sites.
2. Blocks the phishing sites in real-time.

Literature Survey

Machine learning based phishing detection from URLs

Due to the rapid growth of the Internet, users change their preference from traditional shopping to the electronic commerce. Instead of bank/shop robbery, nowadays, criminals try to find their victims in the whether a web page is legitimate or phishing is a very challenging problem, due

cyberspace with some specific tricks. By using the anonymous structure of the Internet, attackers set out new techniques, such as phishing, to deceive victims with the use of false websites to collect their sensitive information such as account IDs, usernames, passwords, etc. Understanding whether a web page is legitimate or phishing is a very challenging problem, due to its semantics-based attack structure, which mainly exploits the computer users' vulnerabilities. Although software companies launch new anti-phishing products, which use blacklists, heuristics, visual and machine learning-based approaches, these products cannot prevent all of the phishing attacks. In this paper, a real-time anti-phishing system, which uses seven different classification algorithms and natural language processing (NLP) based features, is proposed. The system has the following distinguishing properties from other studies in the literature: language independence, use of a huge size of phishing and legitimate data, real-time execution, detection of new websites, independence from third-party services and use of feature-rich classifiers. For measuring the performance of the system, a new dataset is constructed, and the experimental results are tested on it. According to the experimental and comparative results from the implemented classification algorithms, Random Forest algorithm with only NLP based features gives the best performance with the 97.98% accuracy rate for detection of phishing URLs.

Detection and Prevention of Phishing Websites using Machine Learning Approach

Phishing costs Internet user's lots of dollars per year. It refers to exploiting weakness on the user side, which is vulnerable to such attacks. The phishing problem is huge and there does not exist only one solution to minimize all vulnerabilities effectively, thus multiple techniques are implemented. There are three approaches for detecting phishing websites. First is by analyzing various features of URL , second is by checking legitimacy of website by knowing

where the website is being hosted and who are managing it, the third approach uses visual appearance based analysis for checking genuineness of website. And used Machine Learning techniques and algorithms for evaluation of these different features of URL and websites.

Dataset and its Preprocessing

To evaluate the proposed approach, ‘ISCXURL-2016’ dataset is used from the data repository of the University of Canada Brunswick. It contains spam records, phishing links, malware, defacements, and benign URLs [39]. 10000 benign and 9964 phishing URLs are extracted from the repository. To prevent data bias, benign and phishing data is used in nearly equal proportion. The effective preprocessing of the dataset and the selection of the right classifier highly affect the result and accuracy of the machine learning-based approach. Data preprocessing is a data mining technique, which cleans the raw data and delivers more relevant data. Raw data contains numerous numbers of missing values, outliers, and unnecessary features. Without preprocessing, biases and deviation may be found in a dataset that may lead to inaccurate assumptions and results. Preprocessing of data involves many steps, such as data cleaning, integration, reduction, and discretization. Our dataset contains large numbers of infinite and NaN (Not a number) values that are replaced by the mean value using the NumPy library in Python. Once the data gets cleaned, feature scaling is applied to the dataset. Feature scaling is a process of placing data in the same range such that one variable is not dominated by others. The distribution of our data in our dataset is presented in Table 1.1.

Table 1.1

Distribution of data in dataset

| S.No | Feature Name | Min Value | Max Value |
|------|---------------------------------|-----------|-----------|
| 1 | No.of token in domain | 0 | 17 |
| 2 | No.of top level domain | 0 | 5 |
| 3 | Length of an url | 26 | 318 |
| 4 | No.of dots in URL | 1 | 26 |
| 5 | No.of delimiter in domain | 1 | 26 |
| 6 | No.of delimiter in path | 0 | 39 |
| 7 | Length of longest token in path | 0 | 248 |
| 8 | No.of digit in a query | -1 | 103 |
| 9 | Domain length | 0 | 15 |

Drawbacks

1. Phishing detection techniques do suffer **low detection accuracy and high false alarm** especially when novel phishing approaches are introduced.
2. Proposed approach uses Random Forest Algorithm for Classification. Random Forest algorithm may change considerably by a small change in the data.
3. Random Forest algorithm computations may go far more complex compared to other algorithms.

Objectives

To develop a machine learning based phishing detection system that can help users to check the legitimacy and maliciousness of an URL within a minimum amount of time. A mechanism was developed that can extract the feature vectors

from URL whenever a user visits that URL. Afterward, the feature vectors are pre-processed and fed into several machine learning algorithms to validate the legitimacy of an URL.

The other objectives of our proposed approach are described below.

- **Reliable security:** To build the phishing detection approach that produces the minimum number of false positives.
- **Real-time security:** A reliable phishing detection mechanism must block the malicious URL without revealing the credentials to the hackers.
- **Low response time:** Building an anti-phishing approach that can detect phishing URLs as early as possible. The low response time will give hackers comparatively less time to steal the credentials.
- **Detection of new phishing websites:** Building an antiphishing system that outperforms the current blacklisting techniques for phishing detection and can detect new phishing websites in the coming future.
- **Scalability:** Developing a phishing detection approach that can be embedded in a device with constrained resources, i.e., IoT, to the devices powered up by multiple CPUs and GPUs.

Scope of the work

Chapter 1 deals with introduction, literature survey, objectives and scope of the present study.

Chapter 2 briefs about two already existing algorithms, methodology and results. The two existing methods are Machine learning based phishing detection from URLs and Detection and Prevention of Phishing Websites using Machine Learning Approach.

Chapter 3 discusses about a novel approach for phishing URLs detection using lexical based machine learning in a real-time environment and its methodology, results.

Chapter 4 gives the conclusion.

Chapter 2

Existing Methods

2. Machine learning based phishing detection from URLs

Introduction:

Due to the rapid developments of the global networking and communication technologies, lots of our daily life activities such as social networks, electronic banking, e-commerce, etc. are transferred to the cyberspace. The open, anonymous and uncontrolled infrastructure of the Internet enables an excellent platform for cyber-attacks, which presents serious security vulnerabilities not only for networks but also for the standard computer users even for the experienced ones. Although carefulness and experience of the user are important, it is not possible to completely prevent users from falling to the phishing scam. Because, to increase the success of the phishing attacks, attackers also get into consideration about the personality characteristics of the end user especially for deceiving the relatively experienced users. End-user-targeted cyber-attacks cause massive loss of sensitive/personal information and even money for individuals whose total amount can reach billions of dollars in a year.

Methodology:

To detect whether the words in the given/tested URLs are used for fraudulent purposes or not, implemented a Maliciousness Analysis Module (MAM), which mainly focuses on detection of Typo squatting. Typo squatting is also known as URL hijacking, which targets computer users who incorrectly type a website address or link from an email or web page by a web browser such as “Goggle.com” instead of “Google.com”. This module gets a word as input and then analyses it according to the flow chart depicted in the below Fig.2.1. In the previous stages, we have gathered some words in WDM, and they may include some brand names or some keywords. Therefore, in this module, At the same time, the similarity of each word is also calculated with the use of Levenshtein Distance, which is also named as Edit Distance algorithm. This algorithm measures the similarity between two strings by taking into account the deletions, insertions, or substitutions operations in the strings. This algorithm mainly calculates the number of moves, which is needed to convert a source word to the target one. The calculated value gives the distance between the source and target words, and this value is used as the similarity value.

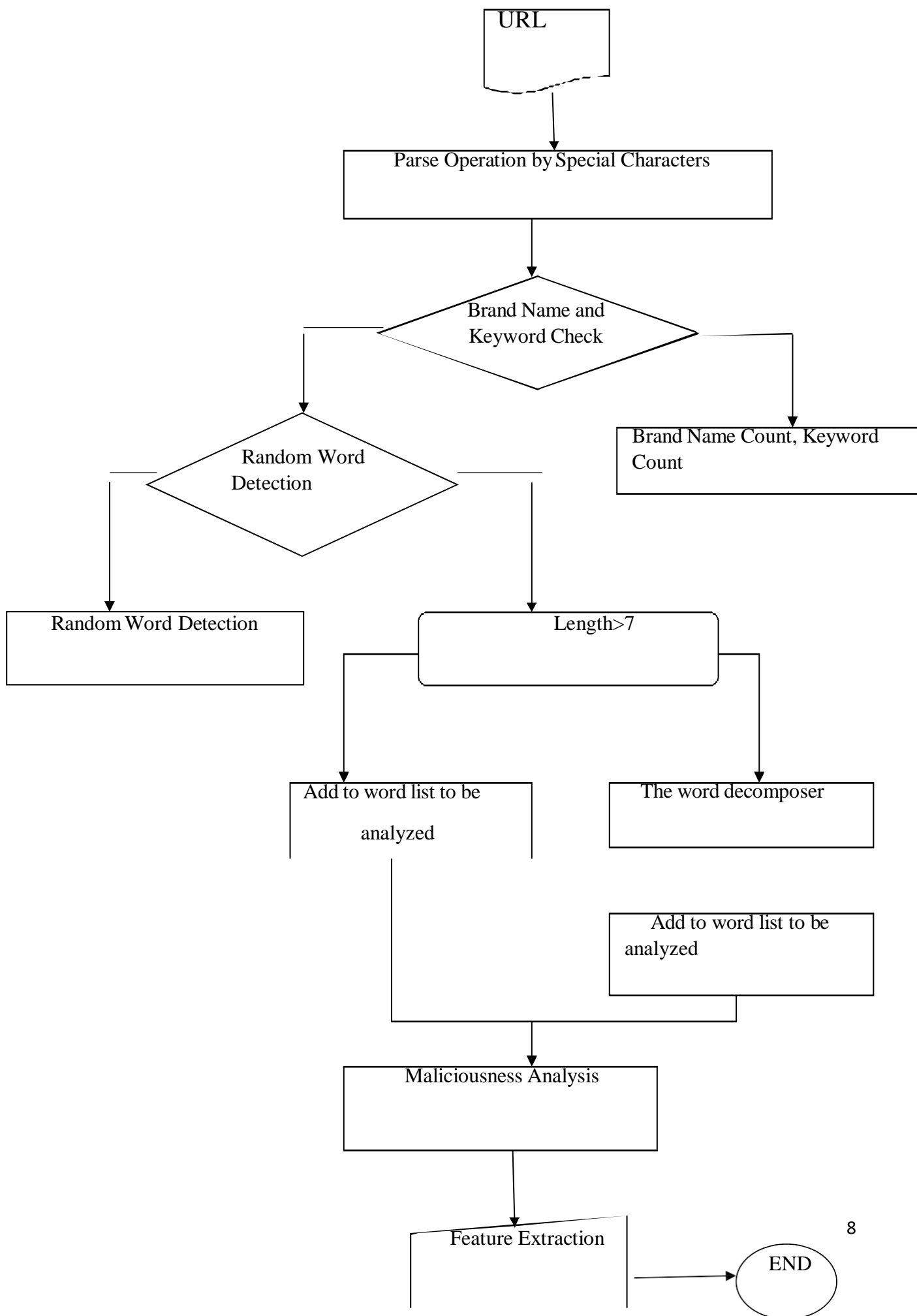


Fig. 2.1 Execution of the data preprocessing mode

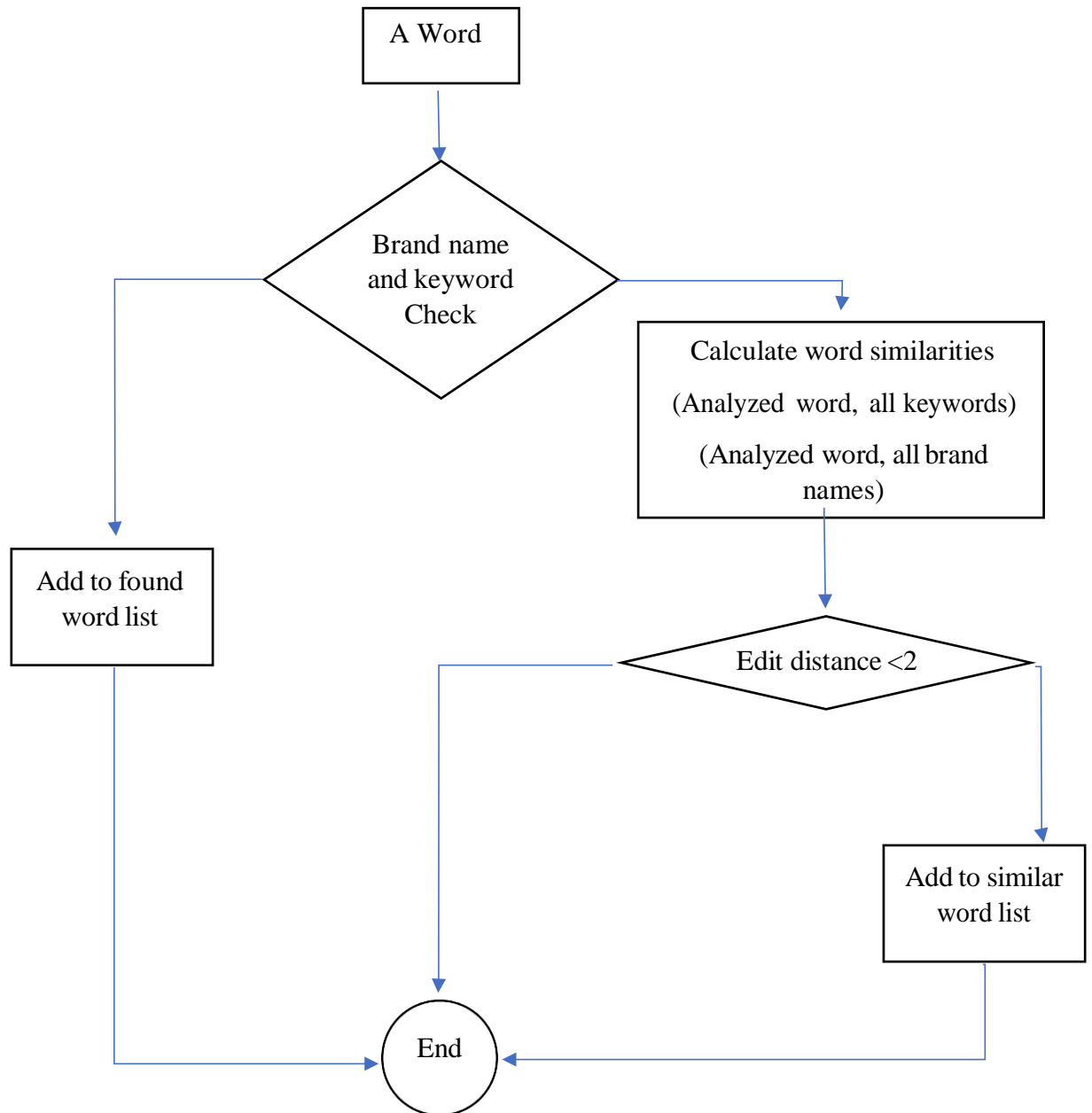


Fig.2.2 Execution of maliciousness analysis

Results:

Table 2.1 : Test results of the classification algorithms.

| Algorithm | Features | Precision | Sensitivity | F-Measure | Accuracy |
|---------------|--------------|-----------|-------------|-----------|---------------|
| Decision Tree | NLP Features | 0.964 | 0.977 | 0.971 | 97.02% |
| | Word Vector | 0.944 | 0.695 | 0.800 | 82.48% |
| | Hybrid | 0.933 | 0.973 | 0.953 | 95.14% |
| Adaboost | NLP Features | 0.908 | 0.963 | 0.935 | 93.24% |
| | Word Vector | 0.936 | 0.536 | 0.682 | 74.74% |
| | Hybrid | 0.915 | 0.940 | 0.927 | 92.53% |
| Kstar NLP | Features | 0.936 | 0.936 | 0.936 | 93.56% |
| | Word Vector | 0.845 | 0.811 | 0.806 | 81.05% |
| | Hybrid | 0.953 | 0.953 | 0.953 | 95.27% |
| KNN(k=3) | NLP Features | 0.940 | 0.977 | 0.958 | 95.67% |
| | Word Vector | 0.955 | 0.697 | 0.806 | 83.01% |
| | Hybrid | 0.946 | 0.974 | 0.960 | 95.86% |
| Random | NLP Features | 0.970 | 0.990 | 0.980 | 97.98% |
| Forest | Word Vector | 0.958 | 0.697 | 0.807 | 83.14% |
| | Hybrid | 0.953 | 0.976 | 0.964 | 96.36% |
| SMO | NLP Features | 0.928 | 0.975 | 0.951 | 94.92% |
| | Word Vector | 0.947 | 0.697 | 0.803 | 82.71% |
| | Hybrid | 0.923 | 0.972 | 0.947 | 94.48% |
| Naïve | NLP Features | 0.940 | 0.977 | 0.958 | 95.67% |
| Bayes | Word Vector | 0.955 | 0.697 | 0.806 | 83.01% |
| | Hybrid | 0.946 | 0.974 | 0.960 | 95.86% |

Conclusion

Implemented a phishing detection system by using seven different machine learning algorithms, as Decision Tree, Adaboost, K-star, KNN ($n = 3$), Random Forest, SMO and Naive Bayes, and different number/types of features as NLP based features, word vectors, and hybrid features. To increase the accuracy of the detection system, construction of an efficient feature list is a crucial task. Therefore, we have grouped our feature list in two different classes as NLP based features, which are mainly human-determined features and word vectors, which focus on the usage of the words in the URL without performing any other operations.

Due to the absence of a worldwide acceptable test set for phishing systems, they needed to construct our own dataset with 73,575 URLs. This set contains 36,400 legitimate URLs and 37,175 phishing URLs. This set is also shared in a web page (Phishing Dataset, 2016) to be used for other phishing detection system developers. According to experimental results, it is clearly seen that the NLP based features have better performance than word vectors with the average rate of 10.86%. Additionally, the use of NLP based features and word vectors together also increases the performance of the phishing detection system with the rate of 2.24% according to NLP based features and 13.14% according to word vectors.

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.

4.3 Support Vector Machine Algorithm [7]

Support vector machine is another powerful algorithm in machine learning technology. In support vector machine algorithm each data item is plotted as a point in n -dimensional space and support vector machine algorithm constructs separating line for classification of two classes, this separating line is well known as hyperplane.

Chapter 3

3. Detection and Prevention of Phishing Websites using Machine Learning Approach

Introduction:

Phishing is a social engineering attack that aims at exploiting the weakness found in the system at the user's end. For example, a system may be technically secure enough for password theft but the unaware user may leak his/her password when the attacker sends a false update password request through forged (phished) website. For addressing this issue, a layer of protection must be added on the user side to address this problem.

A phishing attack is when a criminal sends an email or the url pretending to be someone or something he's not, in order to get sensitive information out of the victim. The victim in regard to his/her curiosity or a sense of urgency, they enter the details, like a username, password, or credit card number, they are likely to acquiesce. The recent example of a Gmail phishing scam that targeted around 1 billion Gmail users worldwide.

Methodology:

Out of all the previous work, only the blacklist and whitelist are implemented which has a drawback of not being updated in long time. The basic idea of our proposed solution is the hybrid solution which uses all the three approaches – blacklist and whitelist, heuristics and visual similarity. Our proposed system has the following algorithm.

1. Monitor all "http" traffic of end-user system by creating a browser extension. The benefit of an extension over an application or software is that the system will be based purely in real time and at the same time will also be quite agile in delivering the outputs.
2. Compare domain of each URL with the white-list of trusted domains and also the black-list of illegitimate domains. The data required for both the lists would be extracted dynamically by web scraping and stored on the server. If domain of the URL is found under the white-list, mark the URL as innocent (Exact Matching), else go further and use the other approaches.

3. Furthermore, the whole website analysis would now be done by considering various details (features). The set of features we took are : website protocol (secure or unsecure), length of the URL, number of hyphen (-) in URL, number of @ symbol in URL, number of dots in the URL, using direct IP address or not, alexa rank, bounce rate, daily page view, whois availability, registration and expiration date of website, alexa.com availability, rank2traffic.com availability, siterankdata.com availability, daily unique visitor, favicon icon similarity and google indexing.
4. Intuitively, the higher similarity between the phishing page and the target page indicates a greater chance of the users being deceived. This is the reason, attackers always try their best to clone the target pages.
5. To counter such antics, our next approach would be to extract and compare CSS of suspicious URL and compare it with the CSS of each of the legitimate domains in queue. This method will look into visual based features of the phished websites.
6. The machine learning classifiers such as decision tree, logistic regression and random forest will be applied to the collected data and a score is generated.
7. Heuristic basic which includes characteristics that are found to exist in phishing attacks in reality and can detect zero-hour phishing attack, but the characteristics are not guaranteed to always exist in such attacks and false positive rate in detection is very high.
8. To overcome the drawbacks of blacklist and heuristics based method, many security researchers now focused on machine learning techniques. Machine learning technology consists of a many algorithms which requires past data .

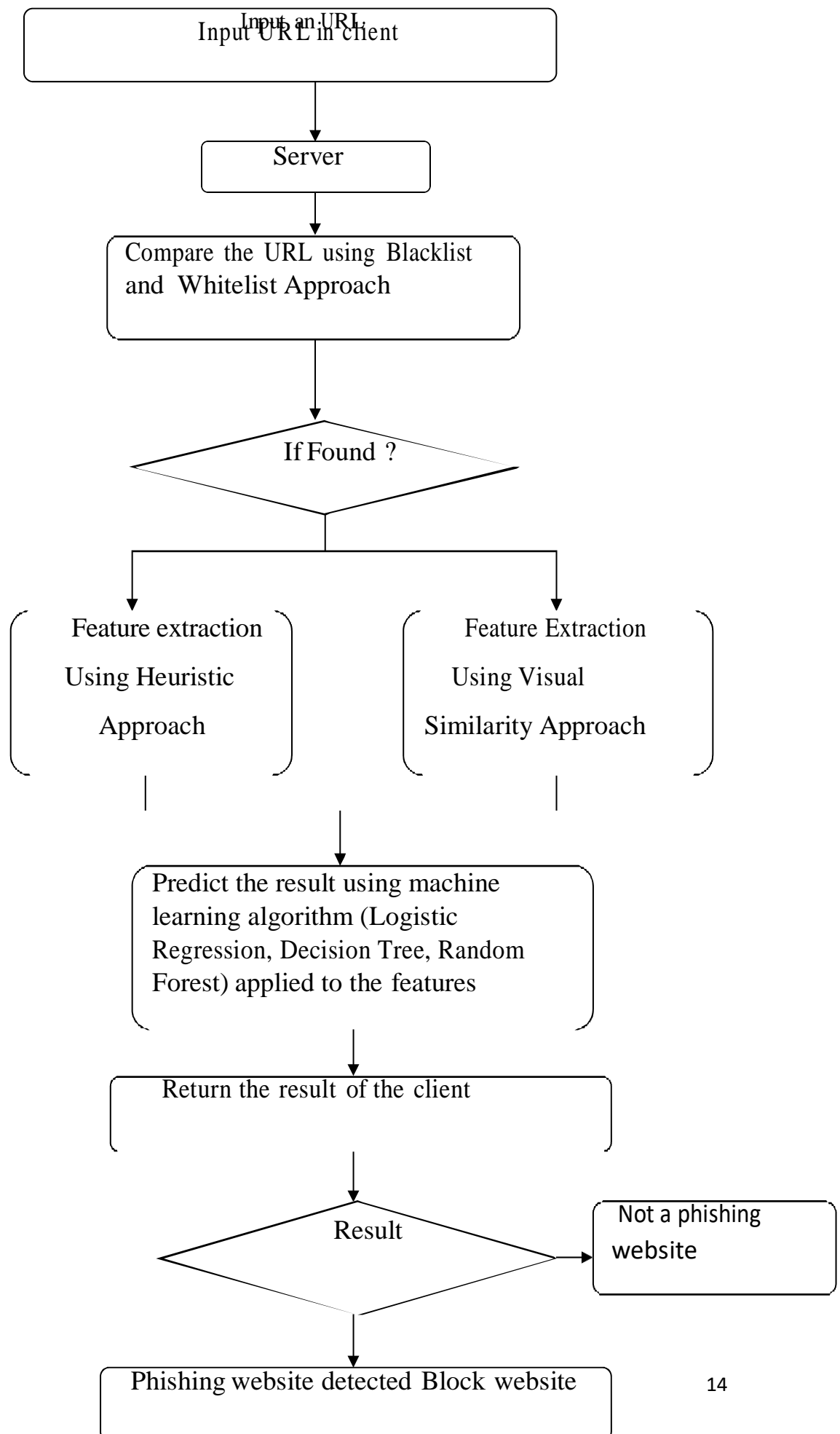


Fig. 2.4 Flow chart of the proposed system

8. The match score and similarity score is calculated. If the score is greater than threshold then we mark the URL as phishing and block it. 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA).

9. This approach basically provides a three level security block and hence can prove to be more effective and accurate than any of the other existing systems.

3.1 Results:

The linear regression plot of expected output versus predicted output is show in Fig.2.5 This was predicted by the random forest algorithm. It has a slight deviation from the expected output for the phished websites.

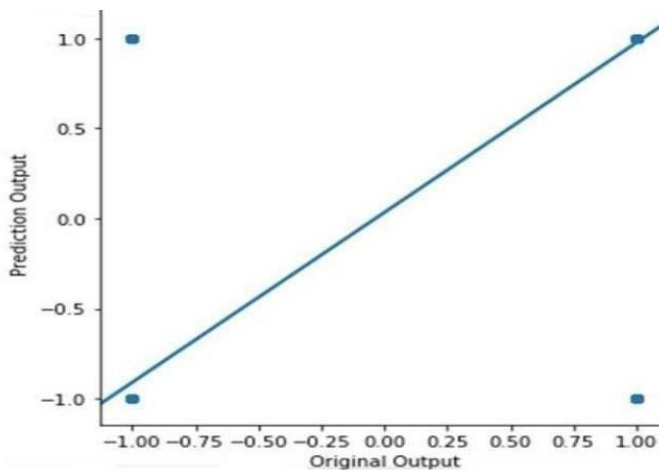


Fig. 2.5 Linear regression plot of original output versus predicted output

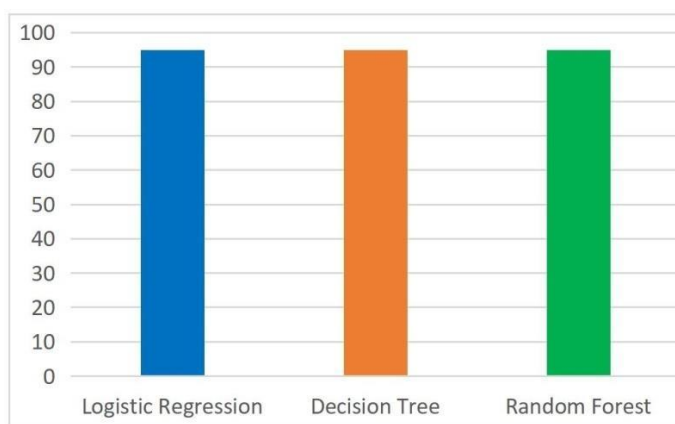


Fig. 2.6 Machine learning accuracy bar plot

The true positive, false positive, true negative, false negative count and accuracy results of 9076 test websites is as shown in Table 2.2.

Table 2.2 : Confusion Matrix Results

| Algorithm | TN | TP | FP | FN | Accuracy |
|---------------------|------|------|-----|----|----------|
| Logistic Regression | 6447 | 2287 | 325 | 17 | 96.23 % |
| Decision Tree | 6393 | 2341 | 326 | 16 | 96.23 % |
| Random Forest | 6392 | 2374 | 297 | 13 | 96.58 % |

3.4 Conclusion

The proposed system enables the internet users to have a safe browsing and safe transactions. Its helps users to save their important private details that should not be leaked. Providing our proposed system to users in the form of extension makes the process of delivering our system much easier. The results points to the efficiency that can be achieved using the hybrid solution of heuristic features, visual features and blacklist and whitelist approach and feeding these features to machine learning algorithms. A particular challenge in this domain is that criminals are constantly making new strategies to counter our defence measures. To succeed in this context, we need algorithms that continually adapt to new examples and features of phishing URL's. And thus, we use online learning algorithms. This new system can be designed to avail maximum accuracy. Using different approaches altogether will enhance the accuracy of the system, providing an efficient protection system. The drawback of this system is detecting of some minimal false positive and false negative results. These drawbacks can be eliminated by introducing much richer feature to feed to the machine learning 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA) algorithm that would result in much higher accuracy.

Chapter 4

PROPOSED MODEL: DETECTION OF PHISHING URLs USING LEXICAL BASED MACHINE LEARNING

Introduction

There has been an exponential growth in the number of organizations, and new technologies are continually being explored for broader applicability. Every day millions of new websites are being developed that have login portals to extract credentials from users. As these websites are in large volume, it is becoming challenging to verify their credibility. As the number of users is increasing, it becomes easy for attackers to lure more users. In most cases, phishing attacks start with fraudulent emails that appear to come from a legitimate source. This email consists of malicious links that are redirected to a fake website when clicked by a user. Afterward, the user ends up giving their confidential information like login id, passwords, and credit card details. Sometimes, attackers perform phishing attacks to disseminate malware in the network.

Modern days phishing is not only limited to email and websites, but also with the links that appear in online ads, status updates, tweets, and Facebook posts. Some of these links are fraudulent where credentials are being massively stolen. Our approach focused on achieving high accuracy with a limited number of features to detect phishing attacks. Thus, studied the most important features in the literature and came up with nine lexical-based features to develop a highly accurate phishing detection approach. Our approach was evaluated with several machine learning classifiers and obtained the highest accuracy of 97.61%.

Methodology

The architecture of our proposed system is shown in Fig. 3.1 In Fig. 3.1, it is observed that whenever a user visits a new URL, the URL is chopped into different segments. Developed a feature extraction mechanism that obtains different lexical based information from URL. Our developed feature extraction mechanism is briefly discussed in Section 3.2.1 Our feature extraction mechanism consists of multiple scripts written in python. The lexical information obtained is then mapped into respective features, and the instance of an URL is prepared that contains all the information necessary to classify the URL as legitimate or phishing. The data from an instance of an URL is then preprocessed, and the preprocessed data is fetched into machine learning algorithms to check the legitimacy of an URL. If the website is found legitimate, then it is given access to the clients to use, otherwise, the website is blocked.

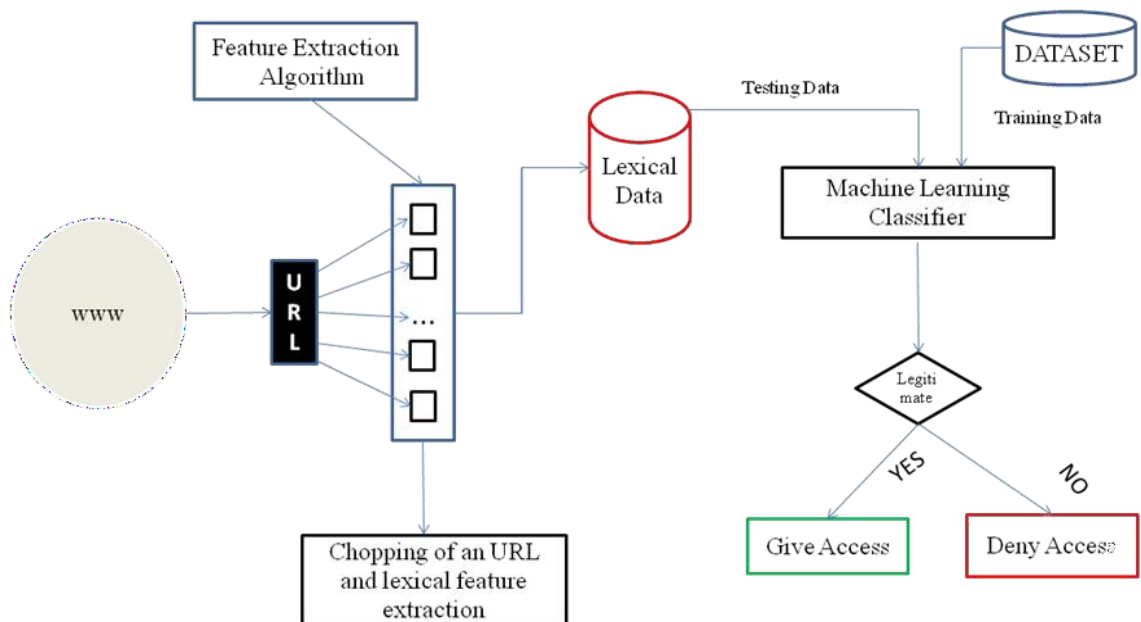


Fig.3.1 Proposed Phishing detection architecture

Features Extraction Algorithm

The most contributing lexical features were developed by analyzing several available lexical features proposed by different researchers. Applied several algorithms to calculate feature importance and came up with the optimal number of features. In our features, we have expanded the number of top-level domains and have included new domains that could present in phishing URLs. Expanding the list has increased the feature importance that can be seen in Fig. 3.2 and ultimately accuracy. The lexical data extracted were then used to make a dataset to train machine learning classifiers.

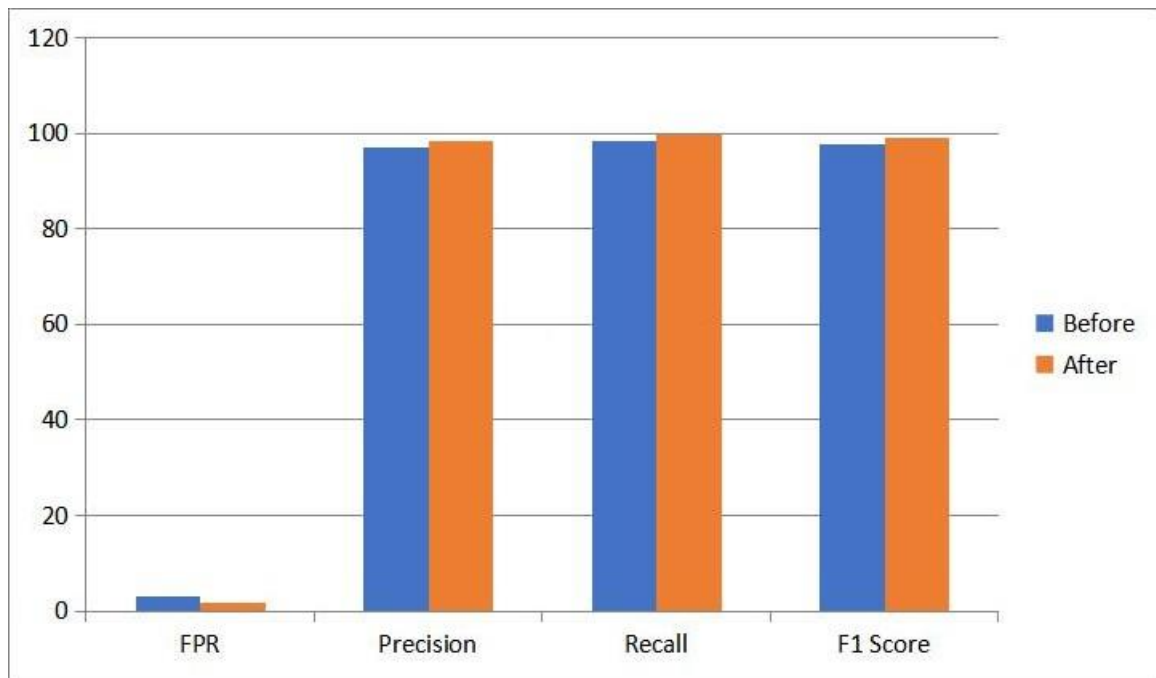


Fig.3.2 Result of SVM Classifier before and after hyperparameter tuning

1. **No.of token in a domain :** Tokenizing is the process of chopping the URL into several pieces. A token refers to a segment that has been broken down into sequence. In this feature, we break down URLs into the tokens and take their count. Generally, it is seen that phishing URLs are longer and contain a greater number of a token count. For e.g., URL: `http://clubeamigosdopedrosegundo.com.br/last/` Tokenized URL=['http', 'clubeamigosdopedrosegundo', 'com', 'br', 'last']. Algorithm 1 gives an idea about the way of tokenizing URLs and taking their count. In algorithm 1, tokenize function helps in creating a token which is stored in a list of `tokenized_list`. In an URL `u`, there might be multiple tokens i.e = { 1 , 2 , 3 , . . . , n } where `T` is a list of `tokenized_list`. The count gives the total number of tokens present in the `T`.

Algorithm 1 : Count the number of tokens in the URL

Input: URL

Output : Number of tokens

Procedure **No.of tokens in URL**

- 2 Initialize count =0
- 3 Initialize index=0
- 4 `Tokenized_list=tokenize(URL)`
- 5 For all each token in `Tokenized_list`:
 `count=count+1`
- 6 Return Count

2. **No. of top-level domain:** It is observed that attackers use multiple top-level domains within a domain name. This feature takes the count of the number of top-level domains present in a URL. Algorithm 2 gives an idea to extract this feature from URLs. In algorithm 2, we have maintained a `top_level_domain` list which contains 920 top-level domains, i.e., = { 1 ,

2 , 3 .. . y }. In one URL, there may be multiple top-level domains present. Let $D = \{d_1, d_2, d_3, \dots, d_n\}$ be the individual top-level domain present in a URL u , such that $d_i \in D$ where d_i is any instance of top level domain present in u . The function `find_top_level_domain` in algorithm 2 searches through the URL to find the number of top level domains. All the domains present in the u are iterated and matched against the top level domain in list D , and the count is stored accordingly.

Algorithm 2 : Find the number of top level domains in an URL

Input: URL

Output: Number of top level domains in URL

Procedure **No.of tld in URL**

1. Initialize count =0
2. Initialize index=0
3. Initialize **T** with all the top level domains
4. Tokenized_list=**tokenise**(URL)
5. For all each token in Tokenized_list:
 - if token in **T**
 - count=count+1
6. Return Count

3. **Length of an URL :** Generally, the phishing URLs are longer in length. Attackers embed multiple domains' names, longer paths to perform malicious activity that makes an URL longer. Algorithm 3 counts each character present inside the length of an URL and gives the total length.

Algorithm 3 : Length of an URL

Input: URL

Output: Length of URL

Procedure **Length_of_URL**

1. Initialise length =0
2. For all each character in URL do
 length=length+1
3. Return length

4. **No.of digits in a query :** To calculate the digit in a query, we first tokenize the text followed by a question mark. In algorithm 4, let us say for URL The function `digits_in_query` iterates through all of the queries present in Q and counts the number of digits present.

Algorithm 4 : Find the number of digits in a query

Input: query

Output: No.of digits in query

Procedure **No_of_digits_in_a_query**

1. Initialise count =0
2. Initialise index=0
3. Initialise digits with {0,1,2,3,4,5,6,7,8,9}
4. For all each var in query:
 if var in digits:
 count=count+1
 index=index+1
5. Return count

5. No. of dots in URL: It is used to check the number of dots present in a URL. Generally, legitimate websites do not have many dots in a URL. On the other hand, phishing websites have comparatively more numbers of dots. Dots are usually used to misguide the legitimacy of the website. For example, www.facebook.com can be represented as www.face.book.com. In such a case, many users do not pay much attention to the URL and become a victim to the attackers.

Algorithm 5 : Find the number of dots in a URL

Input: query

Output: No.of dots in query

Procedure **No_of_dots_in_a_query**

1. Initialise count =0
2. Initialise index=0
3. Initialise dot with ['.']
4. For all each var in query:
 if var in dot:
 count=count+1
 index=index+1
5. Return count

6.No.of delimiter in a domain : Delimiter is the bag of the word containing character like (.), (;), ({}) (|), (+). We have created a list of delimiters containing 26 characters initialized by V in algorithm 6. There may be multiple domains present in one phishing URL. In algorithm 6, the function find_available_domain first finds the number of domains present inside the URL. The function delimiter_count checks the available characters inside the domain against the

bag of word V and gives the count. The count is increased accordingly and stored in D. The presence of a delimiter, for example, a dashed or a + symbol in URL www.face-book.com, www.face+book.com can make a phishing URL look like a legitimate one. The analysis result in Fig. 3.2 indicates that there are relatively more delimiters in a domain of phishing URLs than legitimate URLs.

Algorithm 6 : Find the number of delimiters in domain

Input: Domain

Output: No.of delimiters in domain

Procedure **No_of_delimiters_in_a_domain**

1. Initialise count =0
2. Initialise index=0
3. Initialise delimiters with [(.), (;), ({) (|), (+).....]
4. For all each var in domain :
 5. if var in delimiters:

count=count+1

index=index+1
5. Return count

7.No. of delimiter in the path:We analyzed the URLs and found that the delimiter in the path of the phishing URL is about two times more than in the legitimate URL. This analysis can be found in Fig. 3.2 The phishing URL path contains multiple files like .png, .js followed by a delimiter. These files can be harmful in performing malicious activities. In algorithm 6, the function find_available_path finds the number of paths present in an URL. The function delimiter_count then gives the number of delimiters present inside the path.

Algorithm 7 : Find the number of delimiters in path

Input: Path**Output:** No.of delimiters in pathProcedure **No_of_delimiters_in_a_path**

1. Initialise count =0
2. Initialise index=0
3. Initialise delimiters with [(.), (;), ({) (|), (+).....]
4. For all each var in path:
 if var in delimiters:
 count=count+1
 index=index+1
5. Return count

8. **Length of longest token in the path:** Generally, hackers try to hide some malicious information in the long token of a URL path. Our analysis regarding the longest path in Fig. 3.2 indicates that phishing URLs have a relatively 3X longer path than the legitimate URLs. if the path is not present, then -1 is stored.

Algorithm 8 : Find the length of longest token in the path

Input: Path**Output:** Length of longest tokenProcedure **Length_of_longest_token_in_the_path**

1. Initialise count =0
2. Initialise index=0
3. Tokens=tokenise(path)
4. Sort the list Tokens based on length
5. Return length of Tokens[-1] // last token

9. Domain length: Generally, it is found that hackers use long domains to make a URL look legitimate. A legitimate URL generally contains short domains. Fig. gives this comparative analysis between the domain length present between legitimate and phishing URLs. In this feature, we calculate the domain length of each URL and store them in a dataset.

Algorithm 9 : Find the length of domain

Input: Domain

Output: Length of domain

Procedure **Domain_Length**

1. Initialise length=0
2. Initialise index=0
3. For each var in domain:
 length=length+1
 Index=index+1
4. Return length

Experimental Details

Experiments have been performed on a machine having a core i5 processor with a clock speed of 3.4 GHz, 8 GB RAM, and a 2 GB graphics card. Python has been used for the complete implementation of the proposed approach. In recent times, Python has been very popular for machine learning problems due to the availability of its large amount of libraries, such as Scikit-learn, NumPy, and the flexibility of language itself. Used the following Python libraries to implement our anti-phishing approach.

- **Scikit-learn:** It is used to implement different machine learning algorithms

such as SVM, Random forest, Logistic regression, and KNN.

- **Scipy:** It is used in the statistical analysis of data, such as finding a correlation between the data.
- **Pandas:** It is used to perform data analysis and for data manipulation.
- **Matplotlib:** It is used to find the data distribution and to plot the results in pictorial forms.
- **Time:** It is used to capture the response time of our proposed approach.

The results obtained are evaluated on the basis of the confusion matrix and other metrics that can be found in Table 3.1. Table 3.2 presents all the results generated from different classifiers. Here, we can see that the highest accuracy was offered by Random forest, which is 97.617%. We can see that only six phishing URLs are wrongly classified as legitimate instances out of total phishing instances from the confusion matrix in Table 3.2. The random forest produces the false positive with a rate of 0.53% that renders our proposed approach very reliable and comprehensible. However, a recall score of 97% suggests that we are able to classify 97% of the total instances correctly. In the cases where we are classifying legitimate and phishing URLs, precision matters the most than the recall score.

For performance evaluation, we have to analyze true positive rate (TPR), true negative rate (TNR), false-positive rate (FPR), false-negative rate (FNR), and accuracy. These metrics and their calculation process are clearly described in Table 3.1. Table 3.2 presents the results of these metrics with different classifiers.

Response time is the time between the URL fed, and the result predicted. Several processes are carried out during response time, such as feature extraction, preparation of dataset, loading the module, and predicting the result as phishing or legitimate. The minimal amount of response time is very important to prevent phishing attacks in real-time.

Results and Conclusion

Table 3.1: Performance measure used in proposed approach.

| Performance Measure | Formula | Description |
|---------------------|---|---|
| TPR | $TP/(TP+FN)$ | It is the total number of phishing URLs that are classified as phishing out of total URLs. |
| TNR | $TN/(TN+FP)$ | It is the total number of legitimate URLs that are classified as legitimate out of total URLs. |
| FPR | $FP/(FP+TN)$ | It is the total number of legitimate URLs classified as phishing out of total URLs. |
| FNR | $FN/(FN+TP)$ | It is the total number of phishing URLs classified as legitimate out of total URLs |
| Precision | $TP/(TP+FP)$ | It is the total number of URLs detected as phishing out of total phishing URLs |
| Recall | $TP/(TP+FN)$ | phishing out of total phishing URLs Recall $TP/(TP+FN)$ Total number of legitimate URLs classified as legitimate and phishing URLs classified as phishing |
| F1 Score | $2*(precision*recall)/(precision + recall)$ | The harmonic mean of precision and recall. |
| Accuracy | $(TP+TN)/(TP+TN+FN+FP)$ | Total number of overall correctly classified instances. |

Among all our classifiers, logistic regression performs relatively bad by producing a false positive rate of 3.67%.

Table 3.2: Results from Different Classifiers

| Algorithms | Precision | CM | Recall | F1 Score | Accuracy |
|------------------------|-----------|--------------------------|--------|----------|----------|
| Random Forest | 97 | [2275 51] [59 2226] | 97 | 98 | 97.61 |
| K-Nearest-Neighbor | 94 | [2268 58] [133 2152] | 94 | 96 | 95.85 |
| Support Vector Machine | 89 | [2074 252] [177 2108] | 92 | 91 | 90.69 |
| Logistic Regression | 92 | [2064 262] [172 2113] | 89 | 90 | 90.58 |

Table 3.3: Result of true and false positive classification

| Algorithms | True Positive Rate (%) | False Positive rate (%) | True Negative Rate (%) | False Negative rate (%) |
|------------------------|------------------------|-------------------------|------------------------|-------------------------|
| Random Forest | 99.70 | 0.53 | 99.46 | 0.30 |
| K-Nearest-Neighbor | 98.62 | 0.54 | 99.45 | 1.37 |
| Support Vector Machine | 96.77 | 1.49 | 98.50 | 3.22 |
| Logistic Regression | 94.79 | 3.67 | 96.32 | 5.20 |

In Table 3.4, it is observed that our proposed approach has achieved the highest accuracy among all the existing approaches. A good antiphishing approach must not classify a phishing URL as legitimate. The proposed approach has only 0.53% of FPR, which makes our anti-phishing approach best among all the approaches. Comparative analysis has been shown in Table 3.4.

Table 3.4: Result of different approaches available in a literature

| Approach | FPR(%) | Accuracy(%) | TPI | WPI |
|--------------------------|--------|-------------|-----|-----|
| Sahingoz et al. [16] | 2.95 | 97.98 | Yes | Yes |
| Jain & Gupta et al. [17] | 1.25 | 99.09 | Yes | No |
| Abdelhamid et al. [43] | n/a | 95 | No | No |
| Moghimi et al. [18] | 1.35 | 98.65 | Yes | Yes |
| Chiew et al. [44] | 1.3 | 93.4 | No | No |
| Xiang et al. [45] | 1.4 | 95.8 | No | No |
| Alsarier et al. [48] | 0.0018 | 98 | No | No |
| Zamir et al. [49] | n/a | 97.2 | No | Yes |
| Azeez et al. [50] | n/a | 99.1 | Yes | No |
| Jain & Gupta et al. [51] | 1.52 | 98.4 | Yes | No |
| Our proposed approach | 0.53 | 97.61 | Yes | Yes |

CHAPTER 5

5. IMPLEMENTATION :

```
#from math import log
#from re import compile
from urllib.parse import urlparse
from requests import get
from string import ascii_lowercase
from numpy import array
import re
from collections import Counter
import pandas as pd

class LexicalURLFeature:
    def __init__(self, url):
        self.description = 'blah'
        self.url = url
        self.urlparse = urlparse(self.url)

    # extract lexical features
    def url_scheme(self):
        print(self.url)
        print(self.urlparse)
        return self.urlparse.scheme

    def url_length(self):
        return len(self.url)

    def url_path_length(self):
        return len(self.urlparse.path)

    def url_domain_length(self):
        return len(self.urlparse.netloc)

    def url_netloc(self):
        domain_name = self.urlparse.netloc
        return domain_name
```

```

def url_domain_token_count(self):
    domain_name = self.urlparse.netloc
    delimiters=".",":","/"
    regex_pattern = '|'.join(map(re.escape, delimiters))
    return len(re.split(regex_pattern,domain_name))

def Length_of_longest_token_in_the_path(self):
    domain_name = self.urlparse.path
    delimiters=".",":","/"
    regex_pattern = '|'.join(map(re.escape, delimiters))
    x=re.split(regex_pattern,domain_name)
    L=[]
    for i in x:
        l=len(i)
        L.append(l)
    return max(L)

def number_of_digits_in_url(self):
    digits = [i for i in self.url if i.isdigit()]
    return len(digits)

def number_of_digits_in_query(self):
    if self.urlparse.query:
        digits = [i for i in self.urlparse.query if i.isdigit()]
        return len(digits)
    else:return -1

def number_of_dots_in_url(self):
    aa=self.url
    return aa.count(".")

def get_tld(self): Length=len(self.urlparse.netloc.split('.')[0].split(':')[0]) return Length
def for_Domain_tld(self):
    Length=self.urlparse.netloc.split('.')[0].split(':')[0]
    return Length

def delimiter_Domain_count(self):
    delimiters=list("!@#%$%^&*()_+=-\.:;?/>.<")
    c=list(self.urlparse.netloc)
    d=Counter(c)

```

```

    tot=0
    for i in delimiters:
        tot+=d[i]
    return tot
def delimiter_path_count(self):
    delimiters=list("!@#%$%^&*()_+=-\.:;?/>.<")
    c=list(self.urlparse.path)
    d=Counter(c)
    tot=0
    for i in delimiters:
        tot+=d[i]
    return tot

```

```

url=input("Enter Testing URL: ")
a=LexicalURLFeature(url)

```

```

fetures=[]

```

```

url_length=a.url_length() #3
print("Url length is : ",url_length)

```

```

url_path_length=a.url_path_length()
print("Url path length is : ",url_path_length)

```

```

get_tld=a.get_tld()#2
print("Top level domains in url are : ",get_tld)

```

```

number_of_digits_in_query=a.number_of_digits_in_query()#4
print("Number of digits in query of url is : ",number_of_digits_in_query)

```

```

url_domain_length=a.url_domain_length()#9
print("Length of url domain is : ",url_domain_length)

```

```

url_domain_token_count=a.url_domain_token_count()#1
print("Number of tokens in domain of url : ",url_domain_token_count)

```

```

number_of_dots_in_url=a.number_of_dots_in_url()#5
print("Number of dots in url : ",number_of_dots_in_url)

```

```

delimiter_Domain_count=a.delimiter_Domain_count()#6

```



```

print("Number of delimiters in domain of given url : ",delimiter_Domain_count)

delimiter_path_count=a.delimiter_path_count()#7
print("Number of delimiters in path of given url : ",delimiter_path_count)

Length_of_longest_token_in_the_path=a.Length_of_longest_token_in_the_path()#8
print("Length of the longest token in the path : ",Length_of_longest_token_in_the_path)


fetures.append(url_domain_token_count)
fetures.append(get_tld)
fetures.append(url_length)
fetures.append(number_of_digits_in_query)
fetures.append(number_of_dots_in_url)
fetures.append(delimiter_Domain_count)
fetures.append(delimiter_path_count)
fetures.append(Length_of_longest_token_in_the_path)
fetures.append(url_domain_length)
print(f"URL features are \t{fetures}")
df=pd.read_csv("E:\content/Phishing.csv")

df=df[["domain_token_count","tld","urlLen","Query_DigitCount","NumberofDotsinURL","
delimiter_Domain","delimiter_path","LongestPathTokenLength","domainlength","URL_Ty
pe_obf_Type"]]

#print(df)
#Extracting Independent and dependent Variable
x= df.iloc[:, [0,1,2,3,4,5,6,7,8]].values
y= df.iloc[:, 9].values
#print(x[0],y[0])

# Splitting the dataset into training and test set.
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test= train_test_split(x, y, test_size= 0.3, random_state=0)

#Fitting K-NN classifier to the training set
from sklearn.neighbors import KNeighborsClassifier
classifier1= KNeighborsClassifier(n_neighbors=10, metric='minkowski', p=2 )
classifier1.fit(x_train,y_train)
#for testing

```

```

predict=classifier4.predict([fatures])

if array(predict[0]) == "phishing":
    print('This URL is ' + f"{array(predict[0])}")
else:
    print(f"This URL is {array(predict[0])}") ""

y_pred= classifier4.predict(x_test)

from sklearn.metrics import confusion_matrix,accuracy_score

y_pred1= classifier1.predict(x_test) #knn

y_pred2= classifier2.predict(x_test) #svm

y_pred3= classifier3.predict(x_test) #logistic regression

y_pred4= classifier4.predict(x_test) #random forest

aknn=accuracy_score(y_test, classifier1.predict(x_test))*100
asvm=accuracy_score(y_test, classifier2.predict(x_test)) *100
alr=accuracy_score(y_test, classifier3.predict(x_test)) *100
arf=accuracy_score(y_test, classifier4.predict(x_test))*100

print("Accuracy achieved by KNN Classifier : ",aknn)
print('\n')
print("Accuracy achieved by SVM Classifier : ",asvm)
print('\n')
print("Accuracy achieved by Logistic Regression Classifier : ",alr)
print('\n')
print("Accuracy achieved by Random Forest Classifier : ",arf)
print('\n')

import matplotlib.pyplot as plt
AC=[aknn,asvm,alr,arf]
algo=["KNN","SVM","LR","RF"]

fig=plt.figure()
axi=fig.add_axes([0,0,1,1])
axi.bar(algo,AC)
plt.title("PERFORMANCE MEASURE USING ACCURACY")

```

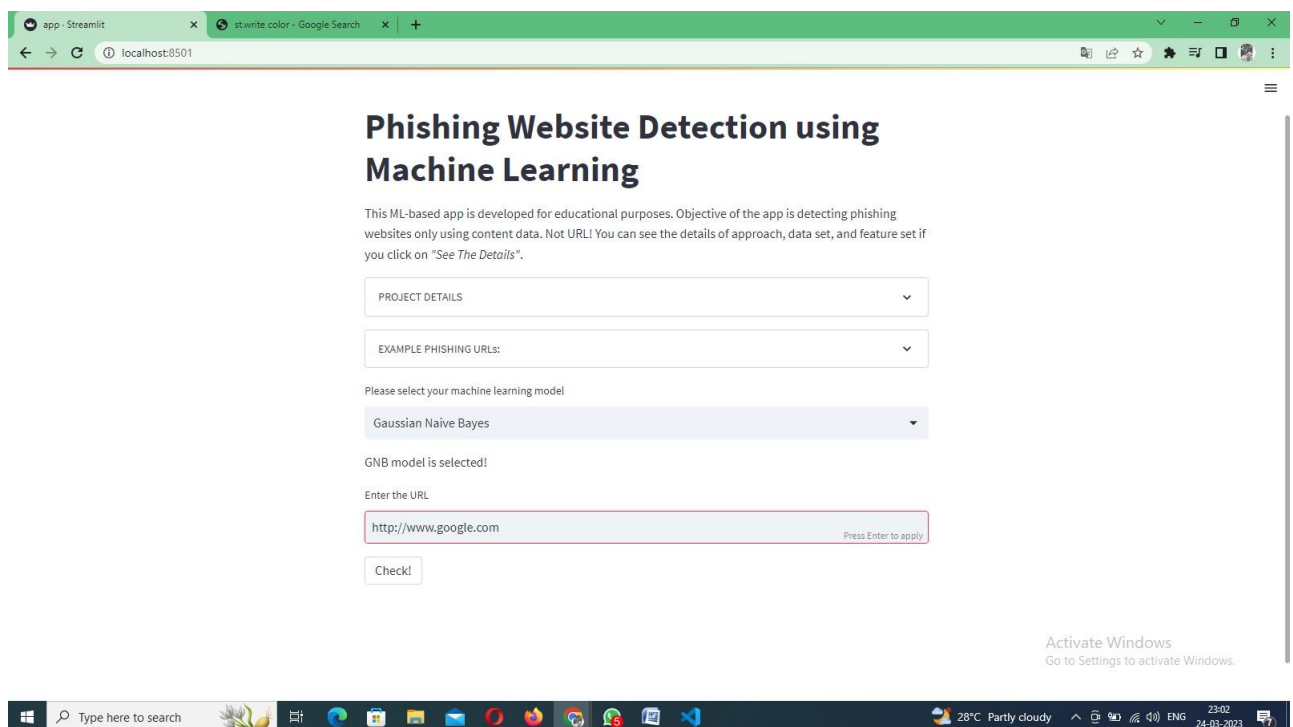
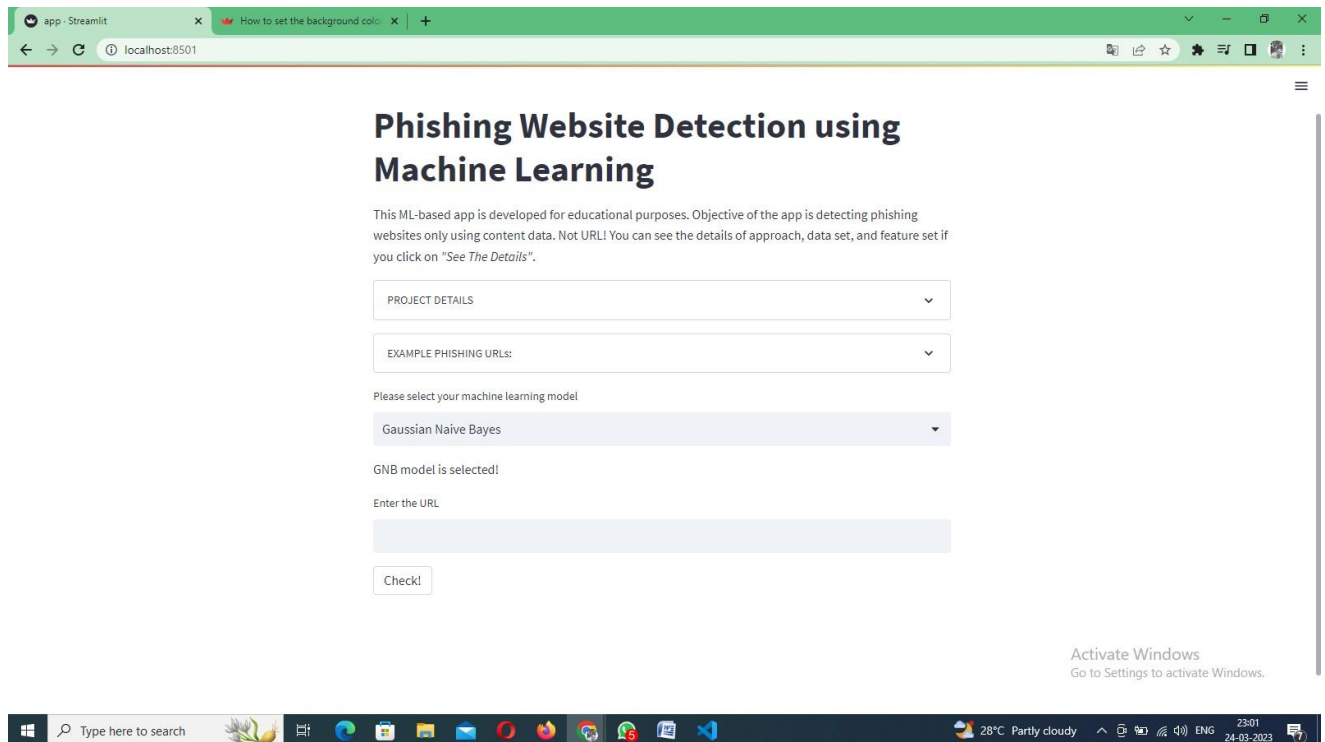
```
plt.show()
```

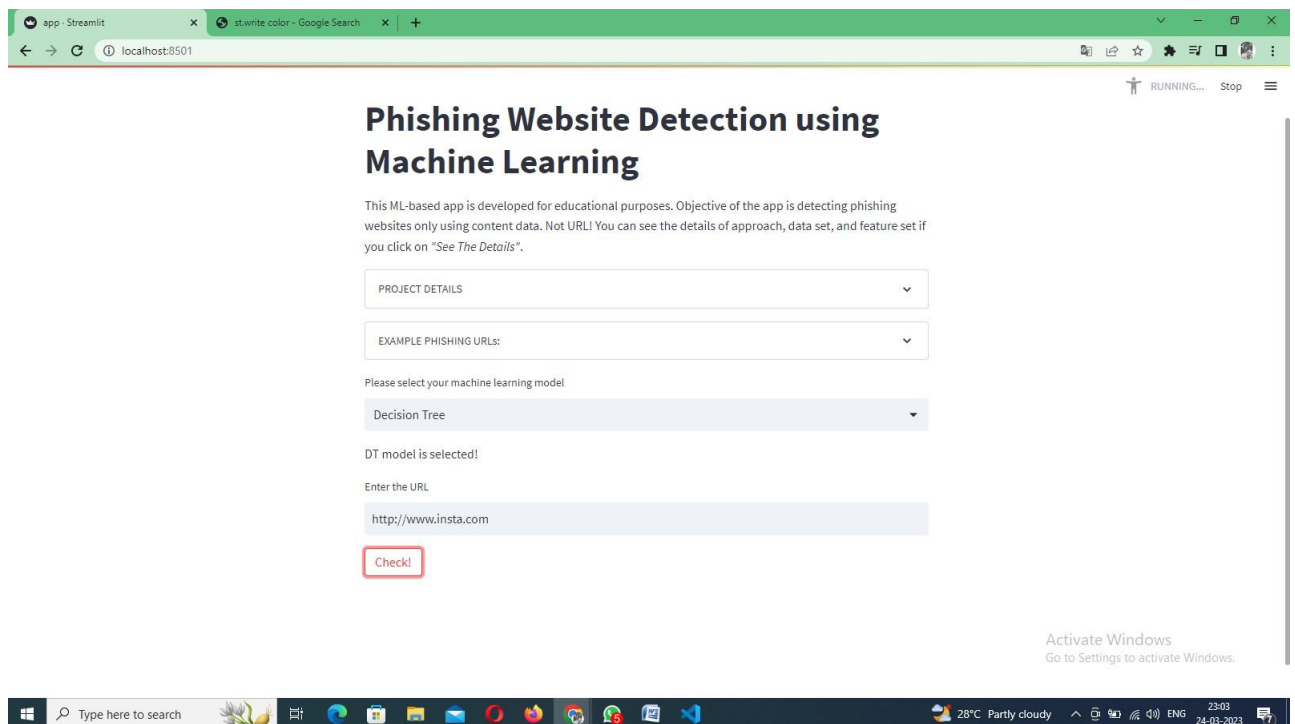
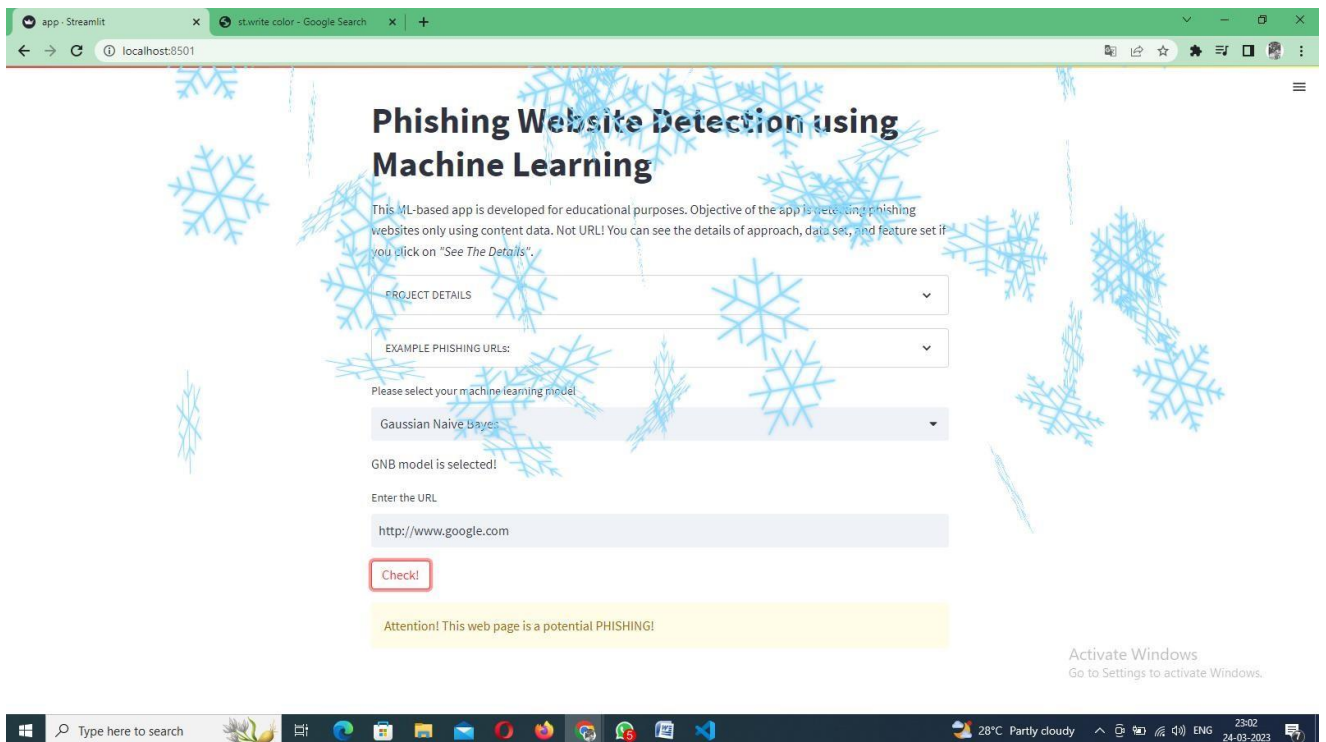
```
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score,  
classification_report
```

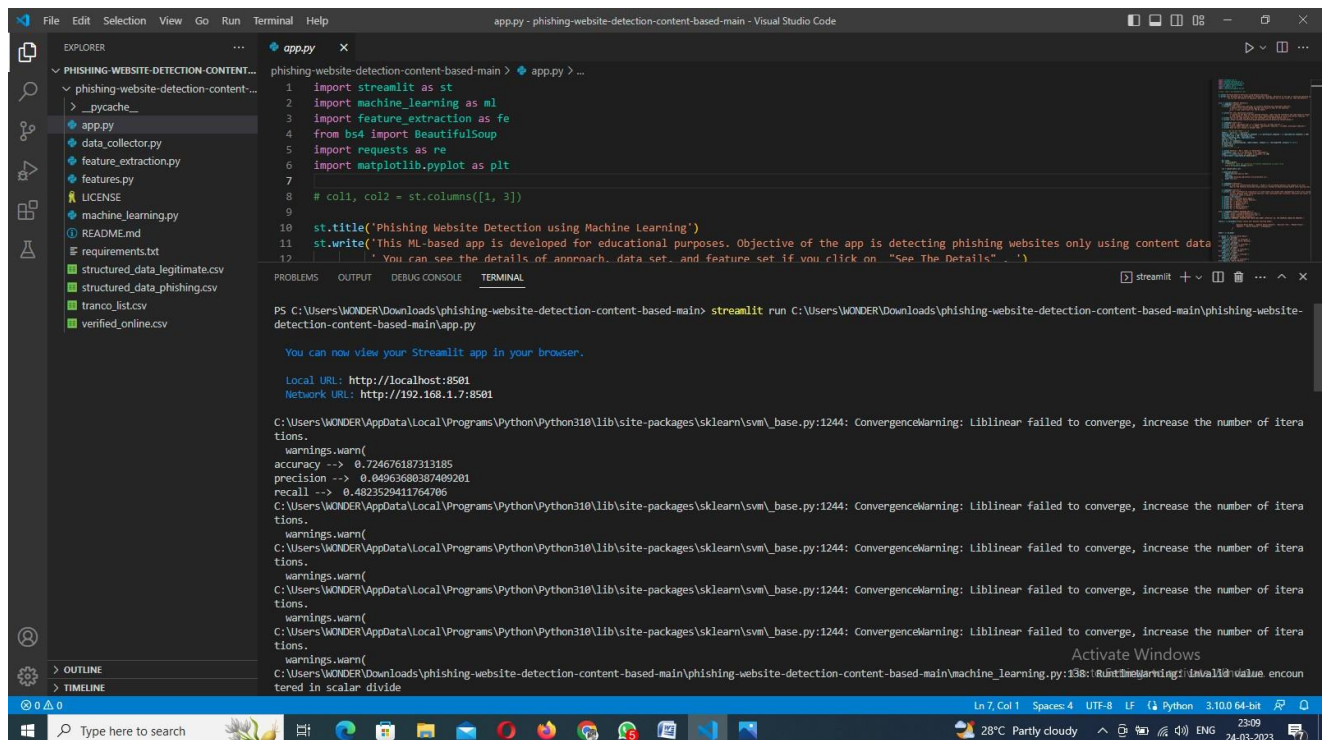
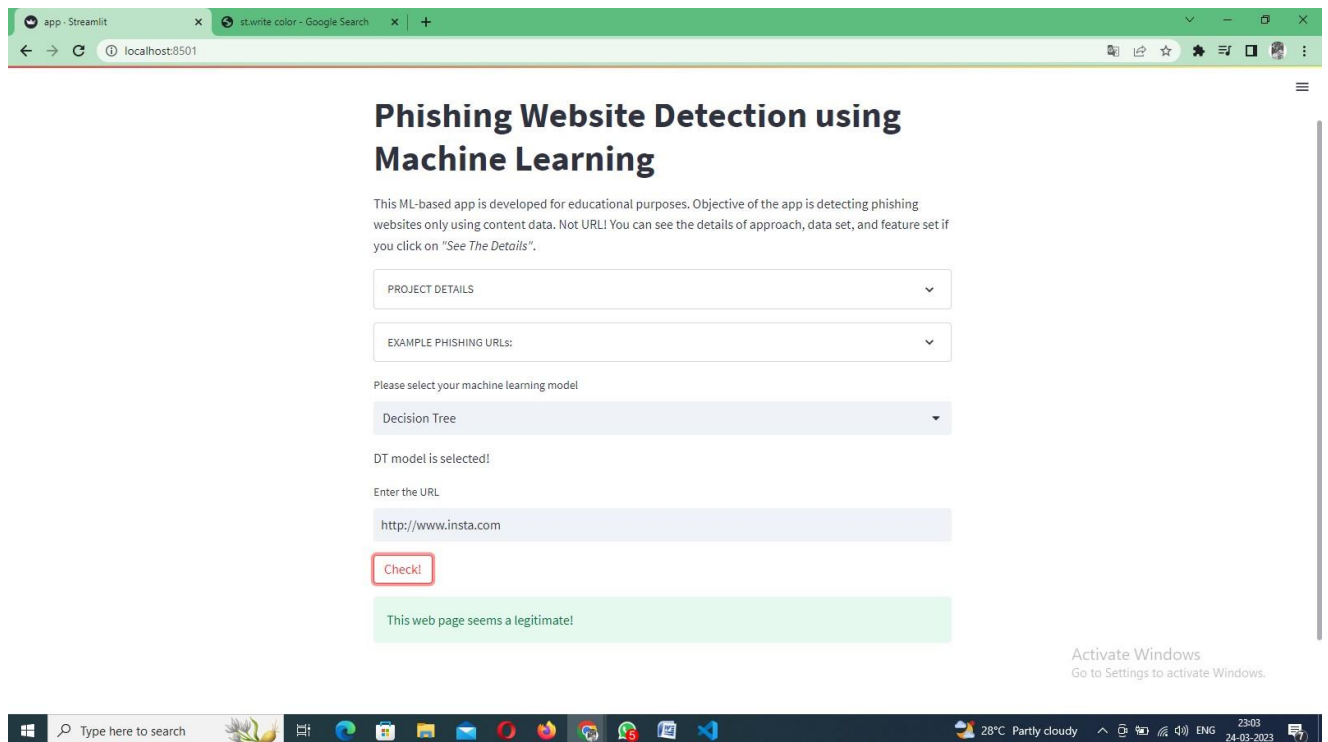
```
def evaluate_model(y_test, y_pred):  
    cm=confusion_matrix(y_test,y_pred)  
    print(cm)  
    print(classification_report(y_test, y_pred))  
    print("\n")
```

```
print("\033[1m+"-----Evaluation report for K Nearest Neighbors -----"+'\033[0m') ;  
print('\n') evaluate_model(y_test,  
y_pred1)  
print("\033[1m+"-----Evaluation report for Support Vector Machine-----  
"+'\033[0m')  
print('\n') evaluate_model(y_test,  
y_pred2)  
print("\033[1m+"-----Evaluation report for Logistic Regression ----- "+'\033[0m')  
print('\n') evaluate_model(y_test,  
y_pred3)  
print("\033[1m+"-----Evaluation report for Random Forest ----- "+'\033[0m')  
print('\n') evaluate_model(y_test,  
y_pred4)
```

6. OUTPUT SCREENS :







7. Experimental Results and Evaluation

Scikit-learn tool has been used to import Machine learning algorithms. Dataset is divided into training set and testing set in 50:50, 70:30 and 90:10 ratios respectively. Each classifier is trained using training set and testing set is used to evaluate performance of classifiers. Performance of classifiers has been evaluated by calculating classifier's accuracy score, false negative rate and false positive rate.

Result shows that Random forest algorithm gives better detection accuracy which is 97.14 with lowest false negative rate than decision tree and support vector machine algorithms. Result also shows that detection accuracy of phishing websites increases as more dataset used as training dataset. All classifiers perform well when 90% of data used as training dataset. Fig. 1 show the detection accuracy of all classifiers when 50%, 70% and 90% of data used as training dataset and graph clearly shows that detection accuracy increases when 90% of data used as training dataset and random forest detection accuracy is maximum than other two classifiers.

8. Conclusion

Started with an overview of phishing attacks and several existing approaches to detect this attack and described the limitation of all the existing approaches and formulated our novel approach in phishing detection. Proposed approach used only nine features based on the lexical properties of URLs and produced an accuracy of 97.61%. To clearly extract the features from URLs, described the feature extraction algorithms in brief. Provided with the results by evaluating our approach with different metrics and brought the detailed comparison between our approach and other phishing detection approaches that have been proposed in the literature. As our future work, will try to evaluate our approach with some sophisticated deep learning algorithms.

9. References

1. Doyen Sahoo, Chenghao Liu, Steven C.H. Hoi, “Malicious URL detection using machine learning: A survey”, 2017, arXiv preprint arXiv:1701.07179.
2. G. Biau, E. Scornet, “A random forest guided tour”, 2016, pp. 197–227.
3. M. Moghimi, A.Y. Varjani, “New rule-based phishing detection method”, *Expert Syst. Appl.* 53 (2016) 231–242.
4. O.K. Sahingoz, E. Buber, O. Demir, B. Diri, “Machine learning base phishing detection from URLs”, *Expert Syst. Appl.* 117 (2019) 345–357.
5. S. Afroz, R. Greenstadt, Phishzoo: “Detecting phishing websites by looking at them”, in: *Fifth IEEE International Conference on Semantic Computing*, 2011.
6. V. Patil, P. Thakkar, C. Shah, T. Bhat, “Detection and prevention of phishing websites using machine learning approach”, in: *Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*, 2018.

PHISHING URL DETECTION

Sk.Chan basha 1, Ch.Sreekanth 2, M.Naga Srikanth 3, G.Saranya 4

1, 2, 3Student, Department of CSE, Narasaraopeta Engineering College, Narasaraopeta, Guntur (D.T)
A.P, India

4Professor, Department of CSE, Narasaraopeta Engineering College, Narasaraopeta, Guntur (D.T)
A.P, India

skchanbasha1111@gmail.com¹, chilamalasrikanth2001@gmail.com², nagasrikanth24@gmail.com³

Abstract

Phishing is a malicious activity that aims to deceive internet users into sharing sensitive information by impersonating legitimate entities. Detecting these URLs is crucial in preventing attacks. The proposed method from the URL and employs a classifier to distinguish between legitimate URLs. The approach was evaluated on a real-world dataset and achieved high accuracy in detecting URLs. The output indicate the approach is effective in detecting phishing URLs and can be used as an essential component in phishing prevention systems. A dataset consisting of malicious and legitimate is for processesing the model. The proposed model achieves an accuracy of 97.5%, outperforming existing state. The results demonstrate machine learning-based approach can effectively detect malicious URLs and can be used as an effective security tool to prevent attacks. The effectiveness of machine learning-based techniques has been demonstrated in several studies, outperforming signature-based and heuristic-based techniques in detecting phishing URLs. These can significantly reduce leading to improve effectiveness in preventing attacks. In summary, learning-based techniques have shown great promise in detecting malicious URLs with high accuracy and future research in this area is likely to yield even better results.

Keywords— Machine learning ,Random forest, CNN, KNN ,Blacklisting, URLs

I. INTRODUCTION

Phishing is a form of cybercrime that involves personal data by impersonating legitimate entities. Phishing attacks are typically carried out through email, social media, or other communication channels that direct users to enter their information into a fake website or form. Phishing attacks can have severe consequences, including identity theft, financial loss, and reputation damage. These URLs can be difficult to detect as they often contain small variations from the original URL or use URL-shortening services to hide the true destination. To prevent attacks, detecting these fraudulent URLs is crucial.

Many methods have been proposed for malicious URL detection, including manual inspection, blacklisting become increasingly popular their learn from and automatically detect patterns that are difficult to identify manually shown great promise in detecting and preventing attacks. By analyzing various features such as domain-based features, content-based features, and lexical-based features, machine learning models can effectively distinguish between phishing and legitimate URLs. In recent years, various machine learning techniques have been proposed for detecting phishing URLs, ranging from traditional classification algorithms to more advanced deep learning approaches. In this paper, we propose a machine learning-based approach for detecting phishing URLs. The proposed model leverages various features and a large dataset of phishing

and legitimate URLs to achieve high accuracy in distinguishing between the two.

II. EXISTING SYSTEM

Over the past 10 years, several techniques have been proposed for detecting phishing URLs, including blacklisting, whitelisting, and other approaches. Let's take a closer look at each of these approaches. Blacklisting is a traditional approach that involves maintaining a database. This approach is based on the assumption that all phishing URLs can be identified and added to the blacklist. However, this approach is not effective against new or unknown phishing attacks, as the database needs to be constantly updated.

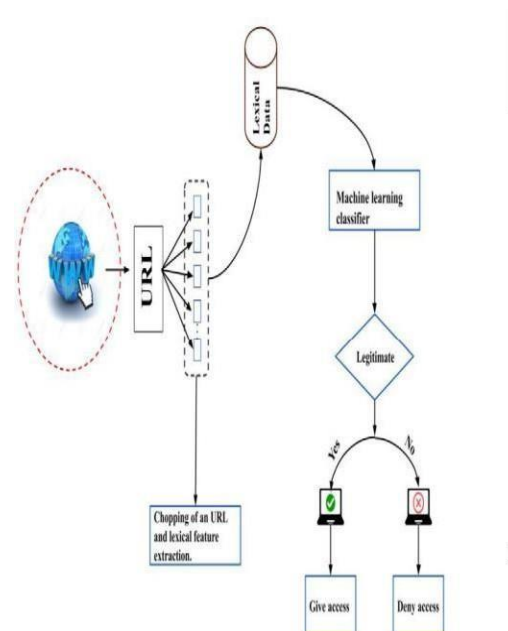
The opposite of blacklisting and involves allowing access only to a predefined list of legitimate URLs. While this approach can prevent access to phishing URLs, it can also block access to legitimate URLs that are not on the This approach is not scalable and requires constant maintenance. Some researchers have proposed hybrid approaches that combine blacklisting techniques with machine learning algorithms. For example, a system may use a blacklist to block known phishing URLs and use machine learning algorithms to detect new or unknown phishing attacks.

This approach can improve the accuracy of phishing URL detection but may require a larger amount of computing resources. Content-based approaches involve analyzing the content of a URL to identify phishing attacks. These approaches may involve analyzing the HTML content, JavaScript code, or other metadata associated with the URL. However, these approaches may be less effective against obfuscated or encrypted content.

III. PROPOSED SYSTEM

A proposed system for phishing URL detection would aim to improve the accuracy and effectiveness of existing systems. One technique would be to extract features presence of suspicious keywords. These features would then be used as inputs. Once the model is trained, it can be used to scan and classify URLs in real-time. Whenever a user clicks on a URL, the system can automatically analyze the URL and compare it with the trained model. If the URL is identified as suspicious, the user can be alerted.

The accuracy percentages of can vary depending on the specific approach and dataset used in the study. However, many studies have reported high accuracy rates for phishing URL detection using ML. For instance, a study conducted by Li and Liang (2017) reported an accuracy rate of 99.72% using a decision tree algorithm for detecting URLs. Another study by Wang et al. (2020) reported an accuracy rate of 99.77% using a support vector machine (SVM) algorithm. Similarly (2019) reported an accuracy rate of 98.52% using a convolutional neural network (CNN) algorithm for detecting phishing URLs. In another (2020) achieved an accuracy rate of 98.68% using an ensemble learning approach for phishing URL detection.



EXPERIMENT ANALYSIS

In this Phishing URL detection using learning techniques often involves analyzing datasets of and legitimate URLs to train learning algorithms to distinguish between the two. These datasets are usually represented in a CSV format, which allows for easy processing and analysis by analyze the dataset, including with an accuracy rate of 98.3%. The decision tree algorithm achieved an accuracy rate of 96.9%, while the random forest and KNN algorithms achieved accuracy rates of 96.7% and 94.6%, respectively. SVM algorithm also outperformed the other algorithms, achieving precision, 98.4%, 98.1%, and 98.2%, respectively. Overall, the experiment demonstrated the effectiveness of machine learning-based approaches for phishing URL detection using CSV datasets. The SVM algorithm proved to be the most effective in this experiment, achieving high accuracy rates. These results highlight the potential of machine learning-based approaches for preventing phishing attacks and protecting against financial losses and privacy breaches. Requirement analysis on phishing URL detection using machine learning (ML) with CSV, involves identifying the key features and functionalities required for an effective phishing URL detection system. The first requirement is to have a dataset Information Phishing URLs often have different DNS information compared to legitimate URLs, such as different IP addresses or name serve. Phishing URLs often have a poor reputation based on their history, IP address, or domain reputation contain keywords such as "login," "bank," "paypal," or "security" to make them appear legitimate. Phishing keywords: Phishing URLs often data, while phishing URLs often do not. These are just a few to created domains, while legitimate URLs are hosted on established domains. Phishing URLs often use domain extensions that are less common or non-standard, such as .tk, .ga, or .ml. Presence

of phishing and legitimate URLs in a CSV format. The dataset should be representative and diverse enough to ensure accurate training and testing of the machine learning algorithms. The system should include data preprocessing and feature extraction functionalities to clean, transform, and normalize the dataset. This step involves identifying and extracting relevant features such as URL length, domain age, and other characteristics that can distinguish phishing from legitimate URLs. The system should include various machine learning algorithms such as decision tree, SVM, KNN, or neural networks to train and test the dataset. The system should also have the ability to fine-tune the algorithms and optimize the hyper parameters for better accuracy, F1 score, and recall. The system should include evaluation metrics such as accuracy, F1 score, and recall to measure the performance of the machine learning algorithms. The system should also include a confusion matrix to visualize. The system should be able to perform real-time phishing URL detection by taking in a URL as input and using the trained machine learning algorithms to determine whether the URL is legitimate or phishing. Feature extraction is a critical step in phishing URL detection using machine learning (ML) as it involves identifying and extracting relevant features from the dataset that can the first step in the process is to collect a dataset of phishing and legitimate URLs. This dataset train the to distinguish phishing legitimate URLs.

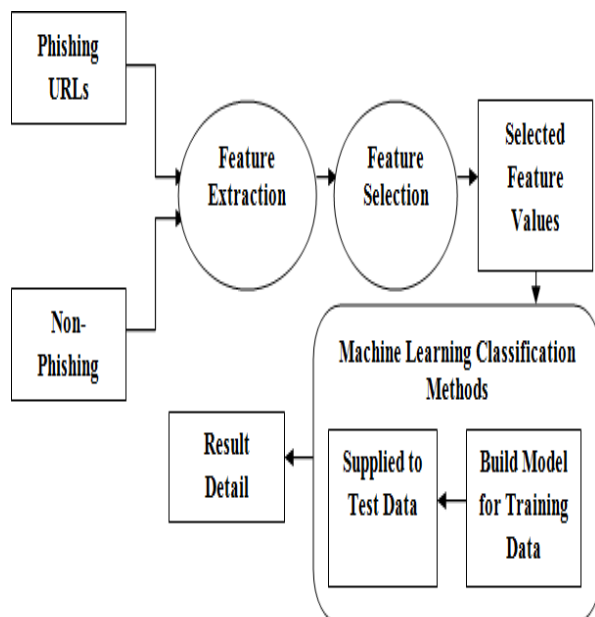
of numbers and special characters Phishing URLs often contain numbers or special characters that are not typically found in legitimate URLs. URL redirects Phishing URLs often use URL redirects to hide the actual URL and make it appear legitimate. Source code analysis Phishing URLs often contain malicious code or scripts that can be identified by analyzing the HTML source code. SSL certificate: Legitimate URLs often use SSL certificates to encrypt extracted for phishing URL detection using ML extracted for phishing URL detection us Feature extraction is a critical step in phishing URL detection using machine learning (ML) as it involves identifying and extracting relevant features from the dataset that can distinguish between phishing and legitimate

URLs. Here are some common features that are used for phishing URL detection using Phishing URLs are often longer than legitimate URLs as they contain additional These are just a few examples of the many

IV. ALGORITHMS

We used for phishing URL detection. Here are some of the commonly used algorithms. Decision trees are simple and effective algorithms by splitting the dataset into smaller subsets based on a series of if-else statements. SVM is a binary classification algorithm that works by finding the optimal boundary between two classes. It can handle high-dimensional datasets and works well with nonlinear data. K-Nearest Neighbors is a lazy learning algorithm that classifies a new data point based on the closest K data points in the training set. It works well for small datasets and can handle both binary and multiclass classification tasks

V. MODEL DESCRIPTION



DATA SELECTION

- Data selection is an important step in building a phishing URL detection system because it helps ensure that the model is trained on high-quality and representative data. Here are some factors to consider when selecting data for phishing URL detection.
- It is important to have a balanced dataset that contains an equal number of phishing and legitimate URLs. This is necessary to avoid bias towards one class and ensure that the model can learn from both types of URLs.
- The dataset should be up-to-date and contain URLs that are currently being used for phishing attacks.

DATA CLEANING

- For instance, URLs with missing data can be removed, and errors in labeling can be corrected.
- Duplicate URLs can bias the model and cause over fitting. Therefore, it is important to remove any duplicate URLs from the dataset.

DATA PRE-PROCESSING AND VISUALIZATIONS

- Data preprocessing is a crucial step in building a phishing URL detection system using machine learning. Here are some common techniques used.
- so that they have similar scales. This is important to ensure that features with larger values do not dominate over

features with smaller values, which can negatively impact model performance.

- creating new features from the existing ones to improve the performance of the model.
- This can include techniques like creating length feature the URL or extracting specific keywords that are commonly used in phishing attacks.
- Feature selection is the process of selecting the most relevant features for the model. This is prevent overfitting.

MACHINE LEARNING TECHNIQUES

- Learning method that builds multiple decision trees and combines their outputs to make a final prediction. Each tree is built on a random subset of the training data, and a random subset of the features is considered at each node of the tree. Random forest can be used for both classification and regression tasks.
- SVM (Support Vector Machine): SVM is a powerful algorithm used for classification and regression tasks. SVM finds the hyper plane best separates data into different classes or predicts value of the variable. SVM works by finding the optimal boundary that maximizes the margin between the different classes.
- KNN (K-Nearest Neighbors): KNN is a simple but effective algorithm used for classification and regression tasks. KNN

predicts the class or value of an unknown data point by looking at the K nearest data points in the training set. The output of KNN depends on the majority class or the mean value of the K nearest neighbors.

- popular algorithm used for binary classification tasks. Logistic regression models the probability of the output variable being in one of the two classes as a function of the input variables. Logistic regression uses a sigmoid function to map the input variables to the probability of the output variable being in one of the classes.

SYSTEM ANALYSIS

- System analysis is an important step in designing an effective phishing URL detection system.
- It involves requirements of the system and analyzing the available resources to determine the best approach for building the system.
- system analysis for phishing URL detection using ML. Problem statement: The first step in system analysis is to clearly define the problem statement..
- In this case, the problem is to develop a system that can accurately detect phishing URLs from legitimate URLs.
- F1 Score: The harmonic mean of precision and recall. It balances both precision and recall and is useful when the classes are imbalanced.
- ROC curve analysis: A graphical representation of the true positive rate. It helps to visualize performance the model across different threshold values.

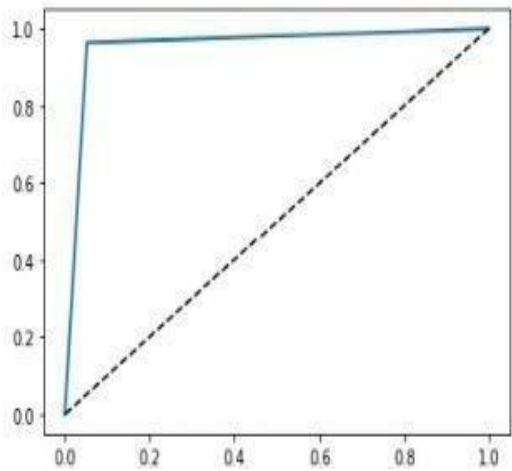


Fig.1. Roc curve for logistic regression

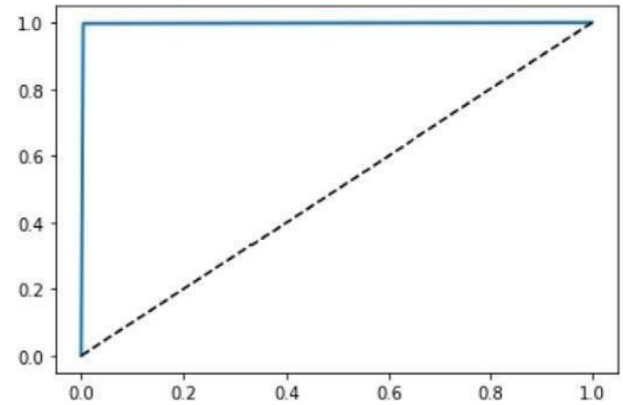


Fig .4. ROC curve for random forest

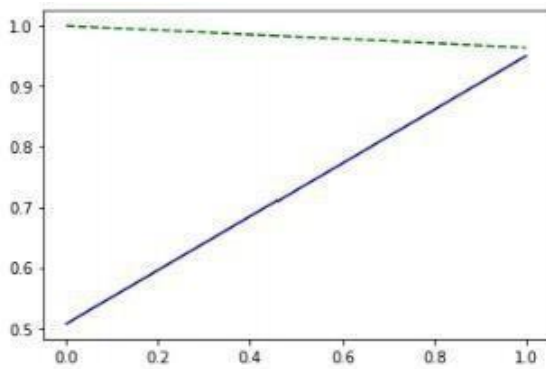


Fig.2.precision-recall trade off logistic regression

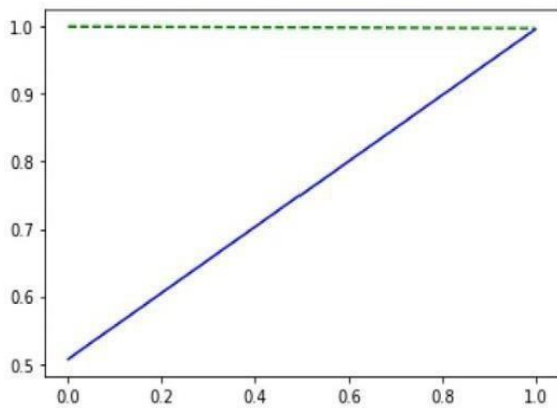
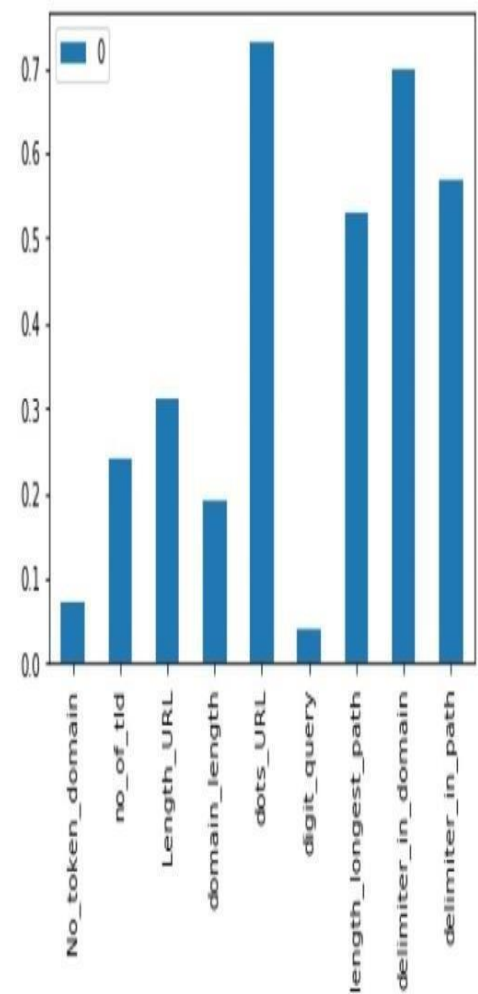


Fig .3.precision-recall trade for random forest



- Accuracy for machine algorithm technique.

| Approach | Description | Advantages |
|------------------------------|---|--|
| Decision Tree | Using decision trees to classify URLs as legitimate or phishing. | Interpretable model, fast training and prediction. Accuracy 92.2% |
| Random Forest | Using random forests to classify URLs as legitimate or phishing. | High accuracy and robustness. Accuracy 96.8% |
| Support Vector Machine (SVM) | Using SVMs to classify URLs as legitimate or phishing. | Good accuracy and generalization. Accuracy 94.3% |
| Logistic Regression | Using logistic regression to classify URLs as legitimate or phishing. | Simple and interpretable model. Accuracy 93.2% |

VI. CONCLUSION

The use of (ML) has shown promise improving accuracy and effectiveness of phishing detection systems. The implementation of a phishing URL detection system using ML involves collecting and preprocessing data, selecting and training an appropriate ML algorithm and ROC curve analysis. The results and analysis of a phishing URL detection system using ML are critical for evaluating the effectiveness of the model and identifying areas for improvement. Continuous improvement and refinement of the model are necessary to ensure that the system .

VII. REFERENCES

- [1] Domain registered report available at: <https://dofo.com/blog/domain-industryreport-april-2020/> Last accessed on May 11, 2020.
- [2] B. Gupta, Amrita Dahiya, Brij A reputation score policy and Bayesian game theory based incentivized mechanism for DDoS attacks mitigation and cyber defense, Future Gener. Comput. Syst. 117 (2021) 193–204.
- [3] S. Atawneh, M. Alauthman, A.A. Almomani, A. Al-Nawasrah, A survey of fast flux botnet detection with fast flux cloud computing, Int. J. Cloud Appl. Comput.(IJCAC) 10 (3) (2020) 17–53.
- [4] M. Ficco, C. Esposito et al, Blockchain-based authentication and authorization for smart city applications, Inf. Process. Manage. 58 (2) (2021) 102468.
- [5] C. Gandhi, S. Kaushik, Ensure hierarchal identity based data security in cloud environment, Int. J. Cloud Appl. Comput. (IJCAC) 9 (4) (2019) 21–36.
- [6] X. Wang, M.K. Khan, Q. Zheng, W. Zhang, et al., A lightweight authenticated encryption scheme based on chaotic scml for railway cloud service, IEEE Access 6 (2017) 711–722.
- [7] A. Dada, O.O. Olakanmi, An efficient privacy-preserving approach for secure verifiable outsourced computing on untrusted platforms, Int. J. Cloud Appl. Comput. (IJCAC) 9 (2) (2019) 79–98

11%

SIMILARITY INDEX

3%

INTERNET SOURCES

8%

PUBLICATIONS

5%

STUDENT PAPERS

PRIMARY SOURCES

| | | |
|---|--|----|
| 1 | Submitted to Coventry University Student Paper | 2% |
| 2 | Brij B. Gupta, Krishna Yadav, Imran Razzak, Konstantinos Psannis, Arcangelo Castiglione, Xiaojun Chang. "A novel approach for phishing URLs detection using lexical based machine learning in a real-time environment", Computer Communications, 2021 Publication | 1% |
| 3 | Submitted to Heriot-Watt University Student Paper | 1% |
| 4 | Submitted to National College of Ireland Student Paper | 1% |
| 5 | "ICCCE 2020", Springer Science and Business Media LLC, 2021 Publication | 1% |
| 6 | Submitted to Liverpool John Moores University Student Paper | 1% |

| | | |
|----|--|------|
| 7 | S. Siva Sunayna, S. N. Thirumala Rao, M. Sireesha. "Chapter 25 Performance Evaluation of Machine Learning Algorithms to Predict Breast Cancer", Springer Science and Business Media LLC, 2022 Publication | 1 % |
| 8 | Thahira A, Ansamma John. "Phishing Website Detection Using LGBM Classifier With URL-Based Lexical Features", 2022 IEEE Silchar Subsection Conference (SILCON), 2022 Publication | 1 % |
| 9 | Submitted to Taylor's Education Group Student Paper | 1 % |
| 10 | medium.com Internet Source | 1 % |
| 11 | Md Fazla Elahe, Min Jin, Pan Zeng. "An Adaptive and Parallel Forecasting Strategy for Short-Term Power Load Based on Second Learning of Error Trend", IEEE Access, 2020 Publication | <1 % |
| 12 | www.researchgate.net Internet Source | <1 % |
| 13 | Graham Williams. "Random Forests", Data Mining with Rattle and R, 2011 Publication | <1 % |
| 14 | ebin.pub Internet Source | <1 % |

15

link.springer.com

Internet Source

<1 %

16

"Adaptive Autonomous Secure Cyber Systems", Springer Science and Business Media LLC, 2020

Publication

<1 %

17

Monika Arora, Vineet Kansal. "Character level embedding with deep convolutional neural network for text normalization of unstructured data for Twitter sentiment analysis", Social Network Analysis and Mining, 2019

Publication

<1 %

18

spectrum.library.concordia.ca

Internet Source

<1 %

Exclude quotes On

Exclude matches Off

Exclude bibliography On

Approved by AICTE, Permanently Affiliated to JNTUK, Kakinada, NIRF Ranking (251-300 Band), Accredited by NBA (Tier-I) & NAAC with 'A+' Grade
Kotappakonda Road, Yellamanda (Post), Narasaraopet - 522601, Palnadu Dist., Andhra Pradesh, INDIA. Website: www.nrtec.in

PAPER ID
NEC/ICAIEA/2K23057

International Conference on
Artificial Intelligence and Its Emerging Areas
NEC-ICAIEA-2K23

17th & 18th March, 2023

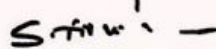
Organized by Department of Computer Science and Engineering in Association with CSI

Certificate of Presentation

This is to Certify that **Shaik Chan Basha** **Narasaraopeta engineering college** has presented the paper title **Phishing url detection** in the International Conference on Artificial Intelligence and Its Emerging Areas-2K23 [NEC-ICAIEA-2K23], Organized by Department of **Computer Science and Engineering in Association with CSI** on 17th and 18th March 2023 at **Narasaraopeta Engineering College, Narasaraopet, A.P., India.**



Convenor
Dr. S. V. N. Srinivasu



Chief-Convenor
Dr. S. N. Tirumala Rao



Principal, Patron
Dr. M. Sreenivasa Kumar



Approved by AICTE, Permanently Affiliated to JNTUK, Kakinada, NIRF Ranking (251-300 Band), Accredited by NBA (Tier-I) & NAAC with 'A+' Grade
Kotappakonda Road, Yellamanda (Post), Narasaraopet - 522601, Palnadu Dist., Andhra Pradesh, INDIA. Website: www.nrtec.in

PAPER ID
NEC/ICAIEA2K23057

Artificial Intelligence and Its Emerging Areas


NEC-ICAIEA-2K23

17th & 18th March, 2023

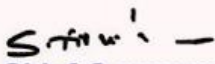
Organized by Department of Computer Science and Engineering in Association with CSI

Certificate of Presentation

This is to Certify that **Chilamala Sreekanth** **Narasaraopeta engineering college** has presented the paper title **Phishing url detection** in the International Conference on Artificial Intelligence and Its Emerging Areas-2K23 [NEC-ICAIEA-2K23], Organized by Department of **Computer Science and Engineering in Association with CSI** on 17th and 18th March 2023 at **Narasaraopeta Engineering College, Narasaraopet, A.P., India.**



Convenor
Dr. S. V. N. Srinivasu



Chief-Convenor
Dr. S. N. Tirumala Rao



Principal, Patron
Dr. M. Sreenivasa Kumar



Approved by AICTE, Permanently Affiliated to JNTUK, Kakinada, NIRF Ranking (251-300 Band), Accredited by NBA (Tier-I) & NAAC with 'A+' Grade
Kotappakonda Road, Yellamanda (Post), Narasaraopet - 522601, Palnadu Dist., Andhra Pradesh, INDIA. Website: www.nrtec.in

International Conference on

PAPER ID
NECICAIEA2K23057

Artificial Intelligence and Its Emerging Areas
NEC-ICAIEA-2K23

17th & 18th March, 2023

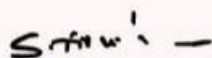
Organized by Department of Computer Science and Engineering in Association with CSI

Certificate of Presentation

This is to Certify that **M NAGA SRIKANTH** Narasaraopeta engineering college has presented the paper title **Phishing url detection** in the International Conference on Artificial Intelligence and Its Emerging Areas-2K23 [NEC-ICAIEA-2K23], Organized by Department of **Computer Science and Engineering** in Association with **CSI** on 17th and 18th March 2023 at **Narasaraopeta Engineering College, Narasaraopet, A.P., India.**



Convenor
Dr. S.V.N. Srinivasu



Chief-Convenor
Dr. S.N. Tirumala Rao



Principal, Patron
Dr. M. Sreenivasa Kumar



Approved by AICTE, Permanently Affiliated to JNTUK, Kakinada, NIRF Ranking (251-300 Band), Accredited by NBA (Tier-I) & NAAC with 'A+' Grade
Kotappakonda Road, Yellamanda (Post), Narasaraopet - 522601, Palnadu Dist., Andhra Pradesh, INDIA. Website: www.nrtec.in

International Conference on

Artificial Intelligence and Its Emerging Areas

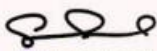
NEC-ICAIEA-2K23

17th & 18th March, 2023

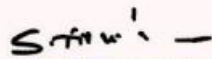
Organized by Department of Computer Science and Engineering in Association with CSI

Certificate of Presentation

This is to Certify that **G Saranya** **Narasaraopeta engineering college** has presented the paper title **Phishing url detection** in the International Conference on Artificial Intelligence and Its Emerging Areas-2K23 [NEC-ICAIEA-2K23], Organized by Department of Computer Science and Engineering in Association with CSI on 17th and 18th March 2023 at Narasaraopeta Engineering College, Narasaraopet, A.P., India.



Convenor
Dr. S. V. N. Srinivasu



Chief-Convenor
Dr. S. N. Tirumala Rao



Principal, Patron
Dr. M. Sreenivasa Kumar

